Fully Secure Attribute Based Encryption from Multilinear Maps

Sanjam Garg* Craig Gentry† Shai Halevi‡ Mark Zhandry§

Abstract

We construct the first fully secure attribute based encryption (ABE) scheme that can handle access control policies expressible as polynomial-size circuits. Previous ABE schemes for general circuits were proved secure only in an unrealistic selective security model, where the adversary is forced to specify its target before seeing the public parameters, and full security could be obtained only by complexity leveraging, where the reduction succeeds only if correctly guesses the adversary's target string x^*, incurring a $2^{|x^*|}$ loss factor in the tightness of the reduction.

At a very high level, our basic ABE scheme is reminiscent of Yao's garbled circuits, with 4 gadgets per gate of the circuit, but where the decrypter in our scheme puts together the appropriate subset of gate gadgets like puzzle pieces by using a cryptographic multilinear map to multiply the pieces together. We use a novel twist of Waters' dual encryption methodology to prove the full security of our scheme. Most importantly, we show how to preserve the delicate information-theoretic argument at the heart of Waters' dual system by enfolding it in an information-theoretic argument similar to that used in Yao's garbled circuits.

1 Introduction

In traditional encryption schemes, access control is all or nothing: the sender encrypts its message under a particular key, and anyone with the corresponding secret key can recover the message. In contrast, attribute-based encryption (ABE) schemes [SW05] allow the sender to embed sophisticated access control policies into its ciphertext. More specifically, an ABE scheme includes an authority, which holds a master secret key and publishes public system parameters, including a relation $R(x, y)$. The sender uses the public parameters to encrypt its message m under some string x to obtain a ciphertext ct_x, where x may (for example) specify some "policy". A user may obtain a secret key sk_y for the string y from the authority (if the authority deems that the user is entitled), where y may specify some "attributes" of the user. If $R(x, y) = 1$, then sk_y can be used to decrypt ct_x to recover m; otherwise decryption fails. In a "ciphertext-policy" ABE scheme [GPSW06], $R(x, y) = x(y)$ – that is, x is viewed as a function or "policy" acting on y. In a "key-policy" ABE scheme, $R(x, y) = y(x)$. In an ABE scheme for general circuits, the policy or relation can be an arbitrary boolean circuit, up to some polynomial bound on size dictated by the public parameters. Using ABE for circuits, a sender can wrap its message in essentially any desired access policy.

Until recently, known ABE schemes could handle only relatively simple relations, and constructing ABE for circuits was the main open problem in the area. Now we have two ABE schemes for circuits, one based on LWE [GVW13] and another based on the multilinear Diffie-Hellman problem

*IBM Research and UC Berkeley, sanjamg@gmail.com

†IBM Research, craigbgentry@gmail.com

‡IBM Research, shaih@alum.mit.edu

§Stanford University, mzhandry@stanford.edu

over multilinear groups [GGH$^+$13c]. However, both of theses schemes have a drawback: they are only *selectively secure* – that is, they have been proved secure only in an unrealistic model in which the adversary is required to specify the string x^* for its challenge ciphertext before it sees the public parameters of the ABE scheme. We would like ABE for circuits that is *fully secure* – i.e., that allows the adversary to choose x^* adaptively after seeing the public parameters and even responses to its private key queries. In general, one can trivially reduce full security to selectively security via *complexity leveraging* – essentially the reduction tries to guess the adversary's chosen x, and succeeds with probability $2^{-|x|}$ – but complexity leveraging loses a $2^{|x|}$ factor in the reduction to the underlying hard problem that we would like to avoid. Achieving full security without the lossiness of complexity leveraging is just as important for ABE for circuits as it was for identity-based encryption (IBE) ten years ago [Wat05, Gen06, Wat09], for both efficiency and conceptual reasons.

1.1 Our Results

We construct the first fully secure ABE for circuits scheme (without complexity leveraging). Our scheme uses n-multilinear maps [BS02, GGH13a, CLT13], where n is proportional to the size of the circuit computing the relation. We base security on fixed, relatively simple assumptions (independent of the relation).

In our scheme, ciphertexts and keys are proportional in size to the circuit computing the relation. It remains an interesting open problem to construct a fully secure ABE for circuits scheme in which the size of the ciphertext is proportional to $|x|$ or the size of the keys is proportional to $|y|$.

1.2 Concurrent and Independent Work

In concurrent and independent work, Waters [Wat14] constructs a fully secure functional encryption (FE) scheme. In a FE scheme, when a ciphertext ct_m that encrypts m is decrypted by a key sk_f for function f, the result is $f(m)$ – that is, the value that is decrypted is a key-dependent function of the message. FE for circuits is more powerful than ABE for circuits, since ABE is the special case where f outputs m iff $R(x, y) = 1$. In this sense, the Waters result is stronger than ours.

However, Waters needs stronger tools and assumptions. In particular, he builds his FE scheme from indistinguishability obfuscation (IO) [GGH$^+$13b]. While tremendous progress has been made on justifying the security of IO [BR14, BGK$^+$14, PST14, GLW14, GLSW14], ultimately the security of the resulting constructions still either relies on an exponential number of assumptions [BR14, BGK$^+$14, PST14] (basically, one per circuit), or a polynomial set of assumptions, but with an exponential loss in the security reduction [GLW14, GLSW14]. For example, the recent IO scheme based on the MSE assumption [GLSW14] crucially uses complexity leveraging in its proof – specifically, the number of hybrids in the proof is proportional to $2^{|x|}$ where x is the input, and each hybrid "examines" a particular input x and implicitly "verifies" that the circuits C_0, C_1 in question satisfy $C_0(x) = C_1(x)$. We note that, using complexity leveraging, it is possible to boost any selectively secure ABE or FE scheme into an adaptively secure scheme.

While Waters' result on FE is exciting, our goal is to build a simpler functionality (fully adaptively secure ABE for circuits) using simpler tools (multilinear maps and simple assumptions involving them) without complexity leveraging.

1.3 Technical Difficulties and New Ideas

1.3.1 The Challenge of Full Security for ABE Schemes

A naive approach for proving the full security of an ABE scheme (or any scheme involving key queries) is the "key-partitioning" strategy. In this approach, the security reduction works by partitioning the y values into two types at the beginning: the set $\mathcal{Y}_{yes}$ for which it knows the secret keys (and for which it can answer key queries), and the complementary set $\mathcal{Y}_{no} = \mathcal{Y} \setminus \mathcal{Y}_{yes}$ for which it doesn't. Let Y_{adv} be the adversary's key queries and x^* be the adversary's value for the challenge ciphertext. For the reduction to succeed, we must have $Y_{adv} \subseteq Y_{yes}$, so that the reduction can answer all of the adversary's key queries, which suggests that the reduction should make Y_{yes} a very dense subset in $\mathcal{Y}$. On the other hand, we must have $R(x^*, y) = 0$ for all $y \in Y_{yes}$, since otherwise (assuming the reduction could efficiently find a $y \in Y_{yes}$ for which $R(x^*, y) = 1$) the reduction could decrypt its own challenge ciphertext without the adversary's help; this suggests that the reduction should take Y_{yes} to be a small subset. Unfortunately, the tension between these two requirements makes it impossible for the key-partitioning strategy to give a tight security reduction. Concretely, consider the relation $R(x, y) = 1$ iff $x \neq y$, and suppose that the adversary makes the single key query $y = x^*$. In this case, the reduction fails unless it correctly guesses x^*, which happens only with probability $2^{-|x|}$.

To prove the full security of ABE with a tight reduction, we need an alternative to the key-partitioning strategy, and our starting point is Waters' dual system methodology [Wat09]. In the dual system, keys and ciphertexts can take on one of two forms: normal or semi-functional. The semi-functional keys and ciphertexts are not used in the real system, but rather only in the proof of security. The reduction always knows how to generate *some* secret key for *any* $y \in \mathcal{Y}$, but (depending on which hybrid in the security proof) the reduction may know how to sample only normal secret keys or only semi-functional keys. The way normal and semi-functional keys and ciphertexts work is that, if ct_x is a ciphertext that encrypts m under string x and sk_y is a key for y and $R(x, y) = 1$, the value of $\text{Decrypt}(ct_x, sk_y)$ equals m *unless* ct_x and sk_y are both semi-functional, in which case the output is uniform (individually over uniform semi-functional ct_x or uniform semi-functional sk_y). In the security proof, the first hybrid corresponds to the real security game, in which the adversary is given normal keys and a normal challenge ciphertext. In the next hybrid, the challenge ciphertext is switched to semi-functional while the keys remain normal. In subsequent hybrids, the keys are switched one-by-one from normal to semi-functional. By the final hybrid, all of the keys are semi-functional and useless for decrypting the semi-functional challenge ciphertext, at which point finishing the security proof becomes relatively easy.

At the heart of the dual system is a subtle information-theoretic argument. Specifically, in a step where a key is switched from normal to semi-functional, the reduction must be prepared to answer a key query for any y and issue a challenge ciphertext for any x. This seems like a paradox, since the reduction could try to determine whether a key sk_y is normal or semi-functional on its own by applying it to a semi-functional ciphertext ct_x for some x satisfying $R(x, y) = 1$. At a high level, known versions of the dual system in the context of (hierarchical) IBE and ABE for simple relations [Wat09, LOS+10, OT10] deal with this problem by ensuring that, when $R(x, y) = 1$, the semi-functional ciphertexts for x and semi-functional keys for y that the reduction can actually efficiently compute are highly correlated, such that decrypting a computable semi-functional ciphertext with a computable semi-functional key actually succeeds – i.e., recovers the intended m. However, the hybrids are structured so that these correlations for x, y satisfying $R(x, y) = 1$ remain *information-theoretically* hidden to the adversary, who is only allowed to query keys for y's satisfying $R(x^*, y) = 0$.

One of the main challenges in using the dual system approach to realize fully secure ABE for circuits is that we need to create a structure that simultaneously allows general computation and preserves some version of the dual system's delicate information-theoretic argument. This new structure, which will be at the heart of our construction, is a new witness encryption scheme [GGSW13].

1.3.2 A Useful Witness Encryption Scheme

A witness encryption scheme is defined for an NP language L with relation $R(x, y)$. A sender can encrypt a message m with respect to a string x to produce a ciphertext ct_x. A recipient can decrypt ct_x to recover m if $x \in L$ and the recipient knows a witness y such that $R(x, y) = 1$. However, if $x \notin L$, then no polynomial-time attacker can distinguish between encryptions of any two equal-length messages. Importantly, the encrypter itself may have no idea whether x is actually in the language. Garg et al. [GGSW13] gave a witness encryption construction for the EXACT COVER problem, and thus for any NP language L.

Conceptually, our new witness encryption scheme is remarkably simple. For each gate g in the circuit computing the relation, the ciphertext includes 4 components associated to the possible assignments $(0, 0, g(0, 0)), \ldots (1, 1, g(1, 1))$ of g's wires. It also includes some components for fixed input x, and components for the free input y. The decrypter who knows a valid witness y can compute the correct assignment of the gates, then use these assignments to pick out the appropriate ciphertext components, and then simply put together the components like puzzle pieces, using the multilinear map as the "glue". Details follow.

For convenience, rather than viewing relations symmetrically, we let C_x be the verification circuit for $x \in L$ – that is, $\mathsf{C}_x(y) = 1$ iff $R(x, y) = 1$. For each wire w in the circuit, let G_w be the set of gates that w feeds into (so that $|G_w|$ is the fan-out of w). Let I be the set of input wires to C_x and W be the set of internal wires of the C_x, including the output wire. Let n, the level of multilinearity, be equal to $|I| + |\mathsf{C}_x|$. Our final ABE construction uses composite order asymmetric graded encodings that can be instantiated with a version of the CLT multilinear maps [CLT13, GLW14], but we start by describing this witness encryption scheme instantiated with ideal *prime order symmetric* multilinear maps, with h_i denoting the generator at the ith level.

Encryption. For $w \in I \cup W, c \in \{0, 1\}$ and $g \in G_w$, sample random values $a_w^c, b_{w,g}^c$ subject to the constraint that $s_w = a_w^0 \prod_{g \in G_w} b_{w,g}^0 = a_w^1 \prod_{g \in G_w} b_{w,g}^1$. For each one of the input wires $i \in I$ we give out $h_1^{a_i^0}$ and $h_1^{a_i^1}$ and for each gate $g \in \mathsf{C}_x$ except for the output gate, we give out four values $h_1^{b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}}$ for $c, c' \in \{0, 1\}$, where $\mathsf{left}(g)$ and $\mathsf{right}(g)$ are the left and right input wires to gate g and $g(c, c')$ is the evaluation of gate g on inputs c and c'. For the output gate we give out only those entries where the output of the gate is actually 1. Finally we encrypt the message M by giving $M \cdot h_n^{\prod_w s_w}$.

Decryption. We are given input y such that $\mathsf{C}_x(y) = 1$. Let y_w be the value assigned to each wire of C_x when evaluated on input y. Note that decryption can be easily done by pairing together the input values $h_1^{a_w^{y_w}}$ for input wires and $h_1^{b^{y_{\mathsf{left}(g)}}_{\mathsf{left}(g),g} b^{y_{\mathsf{right}(g)}}_{\mathsf{right}(g),g} a^{y_{\mathsf{out}(g)}}_{\mathsf{out}(g)}}$ for each gate in the circuit. Note this evaluates exactly to $h_n^{\prod_w s_w}$ which can then be used to recover the encrypted message.

One can view decryption in our WE scheme as being somewhat similar to evaluating a Yao garbled circuit, except that the WE decrypter is allowed to know the underlying assignments to the wires. The gate gadgets in our WE scheme ensure a form of authenticity, but not secrecy (which is unnecessary).

Our WE scheme uses the NP language CircuitSAT or 3SAT, rather than EXACT COVER. One could view our WE scheme as being inspired by the original GGSW scheme, where our new scheme has incorporated the reduction from CircuitSAT or 3SAT to EXACT COVER – e.g., the reduction given by Papadimitriou and Tsitsiklis [PT82]. However, we stress that our scheme cannot be obtained from theirs in a black-box way. It is precisely our "non-black-box" approach of opening open the reduction from CircuitSAT or 3SAT to EXACT COVER that exposes the structure that we need for the dual system's information theoretic argument.

1.3.3 From Witness Encryption toward Attribute Based Encryption

We do not show that our WE scheme has better provably security properties than previous WE schemes. However, projecting our WE scheme down to an ABE scheme does (eventually) give us an ABE scheme with better properties. Some simplifications and observations:

A Fused Input Version of the Witness Encryption Scheme. In the fused input version instead of giving out $h_1^{a_w^0}$ and $h_1^{a_w^1}$ for each input wire $w \in I$, instead we give out one fused value, specifically $h_{|I|}^{\prod_{i \in I} a_i^{y_i}}$. Giving this value allows decryption only if y is a valid witness for $x \in L$. In other words it allows decryption only for a particular valid witness (which will ultimately correspond to an ABE decrypter's string y).

Garbleability of the Fused Input Version. Once an input y to C_x is fixed, this gives a fixed input and intermediate gate values that need to be paired in order to recover the encrypted message. Here we observe that if we were to move to an *asymmetric multilinear maps* then we can in fact garble the other gate entries. The proof for this at an intuitive level is very similar to the proof of Yao's garbed circuit. For each input wire, the value $a_i^{1-y_i}$ is not used as part of the ciphertext. As in Yao's proof, this allows us to garble three out of four gate entries for the gates that only take input wires. Applying this argument recursively we can garble three out of four gate entries for each gate in the circuit. This along with the facts that, (1) $\mathsf{C}_x(y) = 0$ and (2) b_{out}^0 for the output wire out is not used at all; allows us to claim that $\prod_w s_w$ is information theoretically independent of all the other values which are given out. This information theoretic argument is what enables the use of dual system methodology in our context.

1.3.4 An ABE for Circuits Scheme with Partial Adaptivity

As an initial step, we give an ABE construction that only achieves partial adaptivity – i.e., where the adversary is provided with the public parameters before it makes the key queries, but can only make the queries before the challenge ciphertext is provided. For this, we need composite order multilinear maps, where the encodings encode elements from a ring $\mathfrak{R}_1 \times \mathfrak{R}_2$. (The actual construction actually needs three sub-rings for technical reasons similar to those in [LOS+10], but the two sub-ring setting conveys the main ideas well enough for this overview.)

In our scheme, the public parameters, the normal ciphertext and the normal secret-keys all reside in the first sub-ring $\mathfrak{R}_1$, and the second sub-ring will not be used at all[1]. Very roughly, our scheme is essentially the above fused-input WE scheme implemented in the ring $\mathfrak{R}_1$, where a secret key for an input y is a fused-input key for the input y. Decryption is done just as in the fused input witness encryption scheme.

[1] Actually for technical reasons, one component of the secret keys will have a $\mathfrak{R}_2$ component, but this component will be zeroed out during decryption.

In our security proof, just as in regular dual-system proofs, we first switch the challenge ciphertext to semi-functional while keeping the secret-keys normal. The semi-functional ciphertext has additional valid components in the ring $\mathfrak{R}_2$ as well. Next, we switch the queried secret keys one-by-one from normal to semi-functional, which have completely random $\mathfrak{R}_2$ component. Traditional dual systems achieve this by first adding an $\mathfrak{R}_2$ component to the secret key that is correlated with the semi-functional ciphertext. Subsequently, an information theoretic argument is used to remove these correlations and finally move to the semi-functional secret key. In our case, the information-theoretic argument cannot be applied directly because of elaborate correlations that exist between the second dimension of the semi-functional challenge ciphertext and the key being considered. We solve this problem by removing these correlations between the semi-functional ciphertext and key slowly, specifically by using the garbling argument described above. The special property of this garbling process is that it affects only those parts of the semi-functional ciphertext which do not interact with the specific key being considered. Once these unnecessary correlations have been shaved off, we can indeed rely on the information theoretic argument described above to get rid of all the correlations, finally changing the key to its semi-functional form.

Note that, in the steps where we remove the correlations, the changes to the semi-functional ciphertext depend on the attributes of the secret key it is interacting with. For this reason, we only get partially adaptive security, in which the adversary can make key queries only before it gets to see the challenge ciphertext.

1.3.5 Boosting to Full Adaptivity: Mirroring

In our partially adaptive construction, we relied on our ability to temporarily garble a semi-functional ciphertext so that we could change a secret key to its semi-functional form. Our fully adaptive scheme is obtained by stitching this scheme together with its *mirror* image, in which the secret key components become ciphertext components and ciphertext components become secret-key components. This mirror image of our scheme also achieves partial adaptive security, but of the other kind – namely, the adversary is allowed to ask for secret key queries only subsequent to receiving the challenge ciphertext. In the mirror image, the hybrids involve garbling of secrets keys depending on the challenge ciphertext. By carefully stitching these two partially adaptive schemes together, we obtain a fully adaptive scheme. For full details we refer the reader to the scheme itself.

2 Preliminaries

In this section, we start by providing the definition of adaptively secure ABE for general circuits. Next we recall the notion of graded encoding schemes and develop notation that will be needed in our context.

2.1 Adaptively Secure ABE

A ciphertext-policy attribute-based encryption system (more precisely, key encapsulation mechanism) consists of four algorithms: $Setup, KeyGen, Encrypt$, and $Decrypt$.

- **Setup**(λ): The setup algorithm takes in the security parameter λ as input and outputs the public parameters MPK and a master secret key MSK.
- **KeyGen**(MSK, y): The key generation algorithm takes in the master secret key MSK, and an attribute string y as input. It outputs a private key SK_y for y.

- **Encrypt**(MPK, x): The encryption algorithm takes in the public parameters MPK, and an attribute string x as input. It outputs a ciphertext header CT and message encryption key K_{enc} for x. The key K_{enc} is used to encrypt the actual message M. The actual ciphertext consists of the header CT and the encrypted message. We assume that the attribute string x is implicitly included in CT.

- **Decrypt**(SK_y, CT): The decryption algorithm takes a private key SK_y for attribute string y and ciphertext CT for attribute x as input and outputs the message encryption key K_{enc} if $\mathsf{C}(x, y) = 1$. Here C represents a fixed universal circuit.

Correctness of the scheme requires that for correctly generated private keys for y and correctly generated ciphertexts for x, if $\mathsf{C}(x, y) = 1$ then decryption should output the correct message encryption key K_{enc} except with negligible probability.

We will now give the security definition for *adaptive* ABE. This is described by a security game between a challenger and an attacker that proceeds as follows.

- **Setup:** The challenger runs the Setup algorithm and gives the public parameters MPK to the attacker.

- **Query Phase I:** The attacker queries the challenger for private keys corresponding to attribute strings $y_1, \ldots, y_{q_1}$, which the challenger provides.

- **Challenge:** The attacker declares an attribute string x^*. We require that $\forall i \in [q_1]$ we have that $\mathsf{C}(x^*, y_i) = 0$. The challenger runs $(CT, K_{enc}) \leftarrow$ **Encrypt**(MPK, x^*), sets $K_0 = K_{enc}$ and chooses a random $K_1 \in \{0,1\}^\lambda$. The challenger flips a random coin $\beta \in \{0, 1\}$, and gives the pair (CT^*, K_b) to the adversary.

- **Query Phase II:** The attacker queries the challenger for private keys corresponding to the attribute strings $y_{q_1+1}, \ldots, y_q$, with the added restriction that $\forall i \in \{q_1, \ldots, q\}$ we have $\mathsf{C}(x^*, y_i) = 0$.

- **Guess:** The attacker outputs a guess β' for β.

The advantage of an attacker in this game is defined to be $\Pr[\beta = \beta'] - \frac{1}{2}$.

2.2 Graded Encoding Scheme

Now, we describe the graded encoding scheme abstraction that will be needed in our context, mostly following [GGH13a, CLT13, GLW14]. To instantiate the abstraction, we can use Gentry et al.'s variant [GLW14] of the Coron-Lepoint-Tibouchi (CLT) graded encodings [CLT13]. This variant is designed to emulate multilinear groups of composite order, and to allow assumptions regarding subgroups of the multilinear groups.

Definition 1 ($\mathbb{U}$-Graded Encoding System)**.** *A $\mathbb{U}$-Graded Encoding System consists of a ring $\mathfrak{R}$ and a system of sets $\mathcal{S} = \{S_T^{(\alpha)} \subset \{0,1\}^* : \alpha \in \mathfrak{R},\ T \subseteq \mathbb{U}, \}$, with the following properties:*

1. *For every fixed set T, the sets $\{S_T^{(\alpha)} : \alpha \in \mathfrak{R}\}$ are disjoint (hence they form a partition of $S_T \stackrel{\text{def}}{=} \bigcup_\alpha S_T^{(\alpha)}$).*

2. *There is an associative binary operation '+' and a self-inverse unary operation '−' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in \mathfrak{R}$, every set $T \subseteq \mathbb{U}$, and every $u_1 \in S_T^{(\alpha_1)}$ and $u_2 \in S_T^{(\alpha_2)}$, it holds that*

$$u_1 + u_2 \in S_T^{(\alpha_1+\alpha_2)} \quad \text{and} \quad -u_1 \in S_T^{(-\alpha_1)}$$

where $\alpha_1 + \alpha_2$ and $-\alpha_1$ are addition and negation in $\mathfrak{R}$.

3. *There is an associative binary operation '×' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in \mathfrak{R}$, every T_1, T_2 with $T_1 \cup T_2 \subseteq \mathbb{U}$, and every $u_1 \in S_{T_1}^{(\alpha_1)}$ and $u_2 \in S_{T_2}^{(\alpha_2)}$, it holds that $u_1 \times u_2 \in S_{T_1 \cup T_2}^{(\alpha_1 \cdot \alpha_2)}$. Here $\alpha_1 \cdot \alpha_2$ is multiplication in $\mathfrak{R}$, and $T_1 \cup T_2$ is set union.*

CLT (and GGH) encodings do not quite meet the definition of graded encoding systems above, since the homomorphisms required in the definition eventually fail when the "noise" in the encodings becomes too large, analogously to how the homomorphisms may eventually fail in lattice-based homomorphic encryption. However, these noise issues are relatively straightforward (though tedious) to deal with.

Now, we define some procedures for graded encoding schemes.

Instance Generation. The randomized $\mathsf{InstGen}(1^\lambda, \mathbb{U}, r)$ takes as inputs the parameters $\lambda, \mathbb{U}, r$, and outputs params, where params is a description of a $\mathbb{U}$-Graded Encoding System as above for a ring $\mathfrak{R} = \mathfrak{R}_1 \times \ldots \times \mathfrak{R}_r$. We assume $\mathfrak{R}$ is chosen such that the density of zero divisors in each $\mathfrak{R}_i$ is negligible.

Note that setting $r = 1$ corresponds to the prime order setting, while $r > 1$ corresponds to the composite-order setting.

Ring Sampler. The randomized $\mathsf{samp}(\mathsf{params})$ outputs a "level-zero encoding" $a \in S_\phi^{(\alpha)}$ for a nearly uniform element $\alpha \in_R \mathfrak{R}$. (Note that we require that the "plaintext" $\alpha \in \mathfrak{R}$ is nearly uniform, but not that the encoding a is uniform in $S_\phi^{(\alpha)}$.)

Encoding. The (possibly randomized) $\mathsf{enc}(\mathsf{params}, T, a)$ takes a "level-zero" encoding $a \in S_\phi^{(\alpha)}$ for some $\alpha \in \mathfrak{R}$ and index $T \subseteq \mathbb{U}$, and outputs the "level-T" encoding $u \in S_T^{(\alpha)}$ for the same α.

Re-Randomization. The randomized $\mathsf{reRand}(\mathsf{params}, T, u)$ re-randomizes encodings relative to the same index. Specifically, for an index $T \subseteq \mathbb{U}$ and encoding $u \in S_T^{(\alpha)}$, it outputs another encoding $u' \in S_T^{(\alpha)}$. Moreover for any two $u_1, u_2 \in S_T^{(\alpha)}$, the output distributions of $\mathsf{reRand}(\mathsf{params}, T, u_1)$ and $\mathsf{reRand}(\mathsf{params}, T, u_2)$ are statistically indistinguishable.

Addition and negation. Given params and two encodings relative to the same index, $u_1 \in S_T^{(\alpha_1)}$ and $u_2 \in S_T^{(\alpha_2)}$, we have an addition function $\mathsf{add}(\mathsf{params}, T, u_1, u_2) = u_1 + u_2 \in S_T^{(\alpha_1+\alpha_2)}$, and a negation function $\mathsf{neg}(\mathsf{params}, T, u_1) = -u_1 \in S_T^{(-\alpha_1)}$.

Multiplication. For $u_1 \in S_{T_1}^{(\alpha_1)}$, $u_2 \in S_{T_2}^{(\alpha_2)}$ such that $T_1 \cup T_2 \subseteq \mathbb{U}$, we have a multiplication function $\mathsf{mul}(\mathsf{params}, T_1, u_1, T_2, u_2) = u_1 \times u_2 \in S_{T_1 \cup T_2}^{(\alpha_1 \cdot \alpha_2)}$.

Zero-test. The procedure $\mathsf{isZero}(\mathsf{params}, u)$ outputs 1 if $u \in S_\mathbb{U}^{(0)}$ and 0 otherwise. Note that in conjunction with the subtraction procedure, this lets us test if $u_1, u_2 \in S_\mathbb{U}$ encode the same element $\alpha \in \mathfrak{R}$.

Extraction. This procedure extracts a "canonical" and "random" representation of ring elements from their level-$\mathbb{U}$ encoding. Namely $\mathsf{ext}(\mathsf{params}, u)$ outputs (say) $s \in \{0,1\}^\lambda$, such that:

(a) For any $\alpha \in \mathfrak{R}$ and two $u_1, u_2 \in S_{\mathbb{U}}^{(\alpha)}$, $\mathsf{ext}(\mathsf{params}, u_1) = \mathsf{ext}(\mathsf{params}, u_2)$,

(b) For any $i \in [r]$ and $\alpha \in \mathfrak{R}$, the distribution $\{\mathsf{ext}(\mathsf{params}, u) \ : \ \beta \in \mathfrak{R}_i, u \in S_{\mathbb{U}}^{(\alpha+\beta)}\}$ is nearly uniform over $\{0,1\}^\lambda$.

We note that original abstraction of [GGH13a, CLT13] provided uniform distribution only when $\beta \in \mathfrak{R}$. However, the variant in [GLW14] can also be used for the setting when $\beta \in \mathfrak{R}_i$.

3 Our Adaptive ABE Scheme

Notation for circuit C. The scheme uses a universal boolean circuit C with fan-in 2 that is "fixed once and for all". The circuit takes as input (x, y). We will refer to x as the left input and y as the right input. Let I_1 be the set of left input wires to C (input wires corresponding to x) and I_2 be the set of right input wires (input wires corresponding to y). Let W be the set of internal wires of the circuit, including the output wire, referred to as out.

Next we define a set $\mathbb{U}$ of size $2|\mathsf{C}| + |I_1| + 2$. For each gate g in the circuit, we will have two elements in $\mathbb{U}$, namely g and $\overline{g}$. For each left input wire $w \in I_1$, we will also have an element in $\mathbb{U}$, namely w. Additionally we will include $0, 1$ in $\mathbb{U}$.

Instantiating the Graded Encoding System and Our Notation for it. In our construction we will use a $\mathbb{U}$-graded encodings system for a ring $\mathfrak{R} = \mathfrak{R}_1 \times \mathfrak{R}_2 \times \mathfrak{R}_3$. In particular an instance of these encoding can be generated using the procedure $\mathsf{params} \leftarrow \mathsf{InstGen}(1^\lambda, \mathbb{U}, 3)$.

In order to simplify exposition in our scheme we will denote encodings as $[\alpha]_T^1$ where T denotes the level of the encoding and 1 denotes that only the $\mathfrak{R}_1$ component of α is preserved and the component of α in $\mathfrak{R}_2 \times \mathfrak{R}_3$ is zero-ed out. Similarly use of multiple indices in the super-script would be interpreted to suggest that the ring element being encoded has a component in multiple sub-rings. For example $[\alpha]_T^{1,2}$ denotes that only the $\mathfrak{R}_3$ component of α is zero-ed out.

Noise Parameters. We will use two noise parameters σ and σ', with $\sigma' \gg \sigma$. In our scheme all the public parameters will be given out with noise σ and all the ciphertext and secret key encodings will be given out with noise σ'.[2]

Setup(λ, C)**:** Next, for each wire w in the circuit, let G_w be the set of gates that w feeds into (so that $|G_w|$ is the fan-out of w). Then for $c = 0, 1$, $g \in G_w$, sample random $a_w^c, b_{w,g}^c$ from ring $\mathfrak{R}$ subject to the constraint that $s_w = a_w^0 \prod_{g \in G_w} b_{w,g}^0 = a_w^1 \prod_{g \in G_w} b_{w,g}^1$[3]. Similarly, sample barred versions $\overline{a_w^c}, \overline{b_{w,g}^c}$ with the constraint that $\overline{s_w} = \overline{a_w^0} \prod_{g \in G_w} \overline{b_{w,g}^0} = \overline{a_w^1} \prod_{g \in G_w} \overline{b_{w,g}^1}$. Also sample a random $\alpha \in \mathfrak{R}$.

For any gate $g \in \mathsf{C}$, let $\mathsf{left}(g)$ denote the left input wire to g, $\mathsf{right}(g)$ denote the right input wire to g and $\mathsf{out}(g)$ denote the output wire of g. Finally let $g(c, c')$ denote the output of g on

[2] A slightly more sophisticated noise parameter setting could be used for the sake of better efficiency. However we avoid that here for the sake of simplicity.

[3] This can be accomplished by sampling $k = 2|G_w| + 2$ random values $t_1, \ldots, t_k$ from $\mathfrak{R}$ using $\mathsf{samp}(\mathsf{params})$. Then set $a_w^0 = t_1 t_2, b_{w,g_1}^0 = t_3 t_4, b_{w,g_2}^0 = t_5 t_6, \ldots$ and $a_w^1 = t_k t_1, b_{w,g_1}^1 = t_2 t_3, b_{w,g_2}^1 = t_4 t_5, \ldots$.

inputs c and c'. The public key is:

$$MPK = \Big(F = [\alpha]^1_{\mathbb{U}}, G = [1]^1_{\{\overline{g}: g \in \mathsf{C}\} \cup I_1 \cup \{1\}}, \left\{ \overline{E_g^{c,c'}} = \left[\overline{b^c_{\mathsf{left}(g),g}} \cdot \overline{b^{c'}_{\mathsf{right}(g),g}} \cdot \overline{a^{g(c,c')}_{\mathsf{out}(g)}} \right]^1_{\{\overline{g}\}} \right\}_{g \in \mathsf{C}; c,c' \in \{0,1\}},$$
$$\left\{ J_{w,c} = [a^c_w \cdot \overline{a^c_w}]^1_{\{w\}} \right\}_{w \in I_1; c \in \{0,1\}}, B = \left[b^1_{\mathsf{out}} \right]^1_{\{1\}} \Big)$$

We also give out the public parameters params for the graded encoding, as well as randomizers at the following levels in the public key (These will be needed during encryption):

- $\{\overline{g} : g \in \mathsf{C}\} \cup I_1 \cup \{1\}$.
- $I_1 \cup \{1\}$.
- $\{\overline{g}\}$ for every $g \in \mathsf{C}$.

As mentioned earlier, all the public parameters are given out with noise σ. The master secret key MSK consists of all the $a^c_w, b^c_{w,g}, s_w$, their barred versions, and α.

Explanation of terms in the master public key.

- α is a global secret that is only given out at the top level of the encoding, and will be used by the sender to generate the message encryption key for a ciphertext. In order for the receiver to compute the message encryption key, he will need α at a lower level, which is not handed out directly. Secret keys will contain α at a lower level, but masked by other random values. The only way to remove the masking and expose α, and hence the message encryption key, we be to combine the appropriate secret key encodings and ciphertext encodings together.
- $a^c_w, b^c_{w,g}, s_w$, and their barred versions, are used to enforce that the un-masking above can only be carried out on a ciphertext with attribute x by using a secret key with attribute y where $\mathsf{C}(x, y) = 1$. We will associate the un-barred values with secret keys, and the barred values with ciphertexts. In more detail, in order to remove the mask, it will be necessary to compute an encoding of $\prod_{w \in \mathsf{C}} \overline{s_w} s_w$. A ciphertext for attribute x will contain random multiples of the $\overline{E_g^{c,c'}}$ values (that have the form $\overline{b^c_{\mathsf{left}(g),g}} \cdot \overline{b^{c'}_{\mathsf{right}(g),g}} \cdot \overline{a^{g(c,c')}_{\mathsf{out}(g)}}$), as well as an encoding of a random multiple of $\prod_{w \in I_1} \overline{a^{x_w}_w}$. It will also contain a random multiple of B (giving an encoding of b^1_{out}). A secret key for attribute y, meanwhile, will contain a random multiple of $\prod_{w \in I_2} \overline{a^{y_w}_w}$. The only way to combine these components to compute $\prod_{w \in \mathsf{C}} \overline{s_w}$ is to choose the $\overline{E_g^{c,c'}}$ corresponding to the evaluation of C on input x, y (that is, use the $\overline{E_g^{c,c'}}$ where c, c' are the values of the input wires to g on the evaluation of C), and the result of the evaluation must be 1 since we only have an encoding of b^1_{out}. The secret keys symmetrically have components allowing the computation of $\prod_{w \in \mathsf{C}} s_w$.

KeyGen(MSK, y)**:** On input y, sample random values θ and ϕ_w from ring $\mathfrak{R}$ for every $w \in W$. Also sample random values $\zeta_0, \zeta_1, \zeta_{g,c,c'}$ for every $g \in \mathsf{C}; c, c' \in \{0, 1\}$. Output the secret key components as:

$$K_0 = [\alpha]^{1,2}_{\{g:g\in\mathsf{C}\}\cup\{0\}} + \left[\theta \prod_{w\in W}\phi_w \prod_{w\in\mathsf{C}} s_w \overline{s_w}\right]^1_{\{g:g\in\mathsf{C}\}\cup\{0\}} + [\zeta_0]^3_{\{g:g\in\mathsf{C}\}\cup\{0\}}$$

$$K_1 = \left[\theta \prod_{w\in I_2} a_w^{y_w}\overline{a_w^{y_w}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + [\zeta_1]^3_{\{0\}}$$

$$D_g^{c,c'} = \left[\phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^1_{\{g\}} + \left[\zeta_{g,c,c'}\right]^3_{\{g\}} \text{ for } g\in\mathsf{C}; c,c'\in\{0,1\}$$

As mentioned earlier, all the secret key encodings are given out with noise σ'.

Encrypt(MPK, x)**:** On input x, sample random values $\overline{\theta}$ and $\overline{\phi_w}$ for $w\in W$ from the ring $\mathfrak{R}$. Output the ciphertext as:

$$C_0 = \overline{\theta}\prod_{w\in W}\overline{\phi_w}\cdot G \qquad \left(= \left[\overline{\theta}\prod_{w\in W}\overline{\phi_w}\right]^1_{\{\overline{g}:g\in\mathsf{C}\}\cup I_1\cup\{1\}}\right)$$

$$C_1 = \overline{\theta}\cdot B\cdot\prod_{w\in I_1} J_{w,x_w} \qquad \left(= \left[\overline{\theta}\prod_{w\in I_1} a_w^{x_w}\overline{a_w^{x_w}}\cdot\overline{b^1_{\mathsf{out}}}\right]^1_{I_1\cup\{1\}}\right)$$

$$\forall g\in\mathsf{C}, c,c'\in\{0,1\},\ \ \overline{D_g^{c,c'}} = \overline{\phi_{\mathsf{out}(g)}}\cdot\overline{E_g^{c,c'}} \qquad \left(= \left[\overline{\phi_{\mathsf{out}(g)}}\cdot\overline{b^c_{\mathsf{left}(g),g}}\cdot\overline{b^{c'}_{\mathsf{right}(g),g}}\cdot\overline{a^{g(c,c')}_{\mathsf{out}(g)}}\right]^1_{\{\overline{g}\}}\right)$$

The ciphertext header is:

$$CT = \left(C_0, C_1, \left\{\overline{D_g^{c,c'}}\right\}_{g\in\mathsf{C},c,c'\in\{0,1\}}\right)$$

To compute the message encryption key, first compute

$$H = \overline{\theta}\prod_{w\in W}\overline{\phi_w}\cdot F \qquad \left(= \left[\alpha\overline{\theta}\prod_{w\in W}\overline{\phi_w}\right]^1_{\mathbb{U}}\right)$$

Then the message encryption key is: $K_{enc} = \mathsf{ext}(\mathsf{params}, H)$. Here K_{enc} can be used to encrypt the actual message.

Remark 1. *Note that all the encodings given out in the ciphertext can be re-randomized (to noise σ) using the randomizer provided in the public parameters. We do not mention the re-randomization above explicitly, for the sake of simplicity of notation.*

Decrypt(SK_y, C_x)**:** If $C(x,y)=0$ then output $\perp$ and otherwise proceed as follows. In the evaluation of $C(x,y)$, let c_w be the value assigned to wire w. Then compute:

$$H' = C_0K_0 - C_1K_1\prod_{g\in\mathsf{C}} D_g^{c_{\mathsf{left}(g)},c_{\mathsf{right}(g)}}\overline{D_g^{c_{\mathsf{left}(g)},c_{\mathsf{right}(g)}}}$$

and let $K_{enc} = \mathsf{ext}(\mathsf{params}, H')$.

Correctness. To show correctness, it suffices to show that H' computed by the decryption function is equal to H computed during encryption. Indeed,

$$
\begin{aligned}
H' &= C_0 K_0 - C_1 K_1 \prod_{g \in \mathsf{C}} D_g^{c_{\mathsf{left}(g)}, c_{\mathsf{right}(g)}} \overline{D_g^{c_{\mathsf{left}(g)}, c_{\mathsf{right}(g)}}} \\
&= \left[\overline{\theta} \prod_{w \in W} \overline{\phi_w} \left(\alpha + \theta \prod_{w \in W} \phi_w \prod_{w \in \mathsf{C}} s_w \overline{s_w} \right) - \theta \overline{\theta} \prod_{w \in W} (\phi_w \overline{\phi_w}) \prod_{w \in I_1} \left(a_w^{x_w} \overline{a_w^{x_w}} \right) \prod_{w \in I_2} \left(a_w^{y_w} \overline{a_w^{y_w}} \right) b_{\mathsf{out}}^1 \overline{b_{\mathsf{out}}^1} \right. \\
&\qquad \left. \cdot \prod_{g \in \mathsf{C}} \left(b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{c_{\mathsf{out}}(g)}_{\mathsf{out}(g)} \overline{b^c_{\mathsf{left}(g),g}} \, \overline{b^{c'}_{\mathsf{right}(g),g}} \, \overline{a^{c_{\mathsf{out}}(g)}_{\mathsf{out}(g)}} \right) \right]_{\mathbb{U}}^1 \\
&= \left[\alpha \overline{\theta} \prod_{w \in W} \overline{\phi_w} + \theta \overline{\theta} \prod_{w \in W} \phi_w \overline{\phi_w} \left(\prod_{w \in \mathsf{C}} s_w \overline{s_w} - \prod_{w \in \mathsf{C}} \left(a_w^{c_w} \prod_{g \in G_w} b_{w,g}^{c_w} \right) \left(\overline{a_w^{c_w}} \prod_{g \in G_w} \overline{b_{w,g}^{c_w}} \right) \right) \right]_{\mathbb{U}}^1 \\
&= \left[\alpha \overline{\theta} \prod_{w \in W} \overline{\phi_w} \right]_{\mathbb{U}}^1 = H
\end{aligned}
$$

4 Proof of Security

In this section, we show that the construction presented in the previous section is an adaptively secure ABE scheme for general circuits.

4.1 Hardness Assumptions

Given a $\mathbb{U}$-graded encoding for the ring $\mathfrak{R} = \mathfrak{R}_1 \times \mathfrak{R}_2 \times \mathfrak{R}_3$ we will define our hardness assumptions. Let $\mathbb{U}$ be a universe of size n, and let $\mathbb{V}, \mathbb{W}_1, \mathbb{W}_2$ be any partition of $\mathbb{U}$ into disjoint sets. Our assumptions below will be parameterized by $\mathbb{V}$, $\mathbb{W}_1$, and $\mathbb{W}_2$. Let params' consist the parameters params for the graded encodings, plus randomizers for all singleton sets, as well as randomizers for the index sets $\mathbb{U}$, $\mathbb{V}$, $\mathbb{W}_1$, and $\mathbb{W}_1 \cup \mathbb{W}_2$. We will now state our assumptions using notation that was used to describe our scheme.

Definition 2 (Assumption 1)**.** *The following two distributions are indistinguishable:*

$$
\left(\mathsf{params}', \left\{ R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}}, T_i = [t_i]^{1,2}_{\{i\}}, U_i = [u_i]^1_{\{i\}} \right\}_{i \in \mathbb{U}} \right) \text{ and}
$$
$$
\left(\mathsf{params}', \left\{ R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}}, T_i = [t_i]^{1,2}_{\{i\}}, U_i = [u_i]^{1,2}_{\{i\}} \right\}_{i \in \mathbb{U}} \right)
$$

Intuitively the security of the above assumption relies on the fact that the adversary can not isolate any encoding with a $\mathfrak{R}_2$ component alone.

Definition 3 (Assumption 2)**.** *The following two distributions are indistinguishable:*

$$
\left(\mathsf{params}', \left\{ R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}} T_i = [t_i]^{1,2}_{\{i\}} \right\}_{i \in \mathbb{U}}, U = [u]^{2,3}_{\mathbb{V}}, \left\{ V_i = [v_i]^{1,3}_{\{i\}} \right\}_{i \in \mathbb{V}} \right) \text{ and}
$$
$$
\left(\mathsf{params}', \left\{ R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}} T_i = [t_i]^{1,2}_{\{i\}} \right\}_{i \in \mathbb{U}}, U = [u]^{2,3}_{\mathbb{V}}, \left\{ V_i = [v_i]^{1,2,3}_{\{i\}} \right\}_{i \in \mathbb{V}} \right)
$$

Intuitively the security of the above assumption relies on the fact that the adversary can not pair any of the V_i encodings with U. Additionally pairing any V_i encoding with the R_i, S_i and T_i encodings does not help the adversary in any way.

Definition 4 (Assumption 3)**.** *For any $e \in \mathbb{U}$ the following two distributions are indistinguishable:*

$$\left(\mathsf{params}', \left\{R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}}, T_i = [t_i]^2_{\{i\}}\right\}_{i\in\mathbb{U}}, U = [t_e u]^2_{\{e\}}, V = [t_e v]^2_{\{e\}}, W = [t_e uv]^2_{\{e\}}\right) \text{ and}$$
$$\left(\mathsf{params}', \left\{R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}}, T_i = [t_i]^2_{\{i\}}\right\}_{i\in\mathbb{U}}, U = [t_e u]^2_{\{e\}}, V = [t_e v]^2_{\{e\}}, W = [t_e w]^2_{\{e\}}\right)$$

The above assumption is essentially the DDH assumption in $\mathfrak{R}_2$ (or more closely similar to the SXDH assumption in the setting of bilinear groups.).

Remark 2. *As stated, assumptions 1 and 2 are separate assumptions for each partition $\mathbb{V}, \mathbb{W}_1, \mathbb{W}_2$, meaning the number of assumptions is exponential in $n = |\mathbb{U}|$. However, in current graded encoding candidates, different indexes $e \in \mathbb{U}$ are treated identically, meaning the indexes can be re-labeled arbitrarily. Thus, all the exponentially-many assumptions are generated by a polynomial-sized set of assumptions, namely the assumptions parameterized by k, ℓ, m such that $k + \ell + m = n$, where $\mathbb{V} = [1, k]$, $\mathbb{W}_1 = [k + 1, k + \ell]$, and $\mathbb{W}_2 = [k + \ell + 1, n]$. Similarly, assumption 3 can be collapsed to three assumptions for each k, ℓ, m, depending on whether $e \in \mathbb{V}$, $e \in \mathbb{W}_1$, or $e \in \mathbb{W}_2$ (note that we will actually not need the case where $e \in W_1$).*

Moreover, k, ℓ, m will be determined solely by the size of C and the number of left input wires in I_1 (in particular, $k = |\mathsf{C}| + 1$, $\ell = |I_1| + 1$, and $m = |\mathsf{C}|$). Therefore, we can fix a family of universal circuits $\{\mathsf{C}_i\}_{i\in\mathbb{N}}$ to use in our scheme. Then we will need to rely on only a single instance of each of assumptions 1, 2, and two instances of assumption 3 to argue security.

Remark 3. *We state our assumptions with the R_i being encodings of 1 in the ring $\mathfrak{R}_1$, rather than a random element from $\mathfrak{R}_1$ (as is typically the case in dual system literature). This was mostly for convenience in our security proofs — all our proofs still carry though if the R_i were encodings of random $\mathfrak{R}_1$ elements.*

Remark 4. *Even though we do not explicitly mention this above, we assume that all the encodings in the assumptions are provided with noise parameter σ'' that is much less than the parameters σ and σ', which correspond to the level of noise in the encodings given out in the scheme. These parameters are set such that all the encodings in the proof can be given out according to the correct noise parameters.*

4.2 Additional Structures Needed for the Proof

Before we give our proof of security, we define two additional structures needed for the proof: namely, semi-functional ciphertexts and keys. These will not be used in the real system, but will be needed in our proof.

Semi-functional Ciphertext. A semi-functional ciphertext on input (MPK, x) has the same form as a real ciphertext for (MPK, x), except that a semi-functional ciphertext has components in $\mathfrak{R}_1$ and $\mathfrak{R}_2$, rather than just $\mathfrak{R}_1$.

$$C_0 = \left[\overline{\theta} \prod_{w \in W} \overline{\phi_w}\right]^{1,2}_{\{\overline{g}: g \in \mathsf{C}\} \cup I_1 \cup \{1\}}$$

$$C_1 = \left[\overline{\theta} \prod_{w \in I_1} a_w^{x_w} \overline{a_w^{x_w}} \overline{b^1_{\mathsf{out}}}\right]^{1,2}_{I_1 \cup \{1\}}$$

$$\overline{D_g^{c,c'}} = \left[\overline{\phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g}} \overline{b^{c'}_{\mathsf{right}(g),g}} \overline{a^{g(c,c')}_{\mathsf{out}(g)}}\right]^{1,2}_{\{\overline{g}\}} \quad \forall g \in \mathsf{C}, c, c' \in \{0,1\}$$

The ciphertext is

$$CT = \left(C_0, C_1, \left\{\overline{D_g^{c,c'}}\right\}_{g \in \mathsf{C}, c, c' \in \{0,1\}}\right)$$

The message encryption key is computed as:

$$H = \left[\alpha \overline{\theta} \prod_{w \in W} \overline{\phi_w}\right]^{1,2}_{\mathbb{U}}$$

$$K_{enc} = \mathsf{ext}(\mathsf{params}, H)$$

Semi-functional Key. A semi-functional key will take on one of the following forms.

- A semi-functional key of type 1 has a random $\mathfrak{R}_2$ component in K_0, while all the other secret key encodings are zero in $\mathfrak{R}_2$. In other words, it has the following form:

$$K_0 = \left[\alpha + \theta \prod_{w \in W} \phi_w \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^1_{\{g: g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0]^{2,3}_{\{g: g \in \mathsf{C}\} \cup \{0\}}$$

$$K_1 = \left[\theta \prod_{w \in I_2} a_w^{y_w} \overline{a_w^{y_w}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + [\zeta_1]^3_{\{0\}}$$

$$D_g^{c,c'} = \left[\phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^1_{\{g\}} + \left[\zeta_{g,c,c'}\right]^3_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

- A semi-functional key of type 2 has a random $\mathfrak{R}_2$ component in all secret key encodings, except that the components in $\mathfrak{R}_2$ are correlated across each of the encodings in the secret key. Specifically, it is set as:

$$K_0 = \left[\alpha + \theta \prod_{w \in W} \phi_w \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^{1,2}_{\{g: g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0]^3_{\{g: g \in \mathsf{C}\} \cup \{0\}}$$

$$K_1 = \left[\theta \prod_{w \in I_2} a_w^{y_w} \overline{a_w^{y_w}} b^1_{\mathsf{out}}\right]^{1,2}_{\{0\}} + [\zeta_1]^3_{\{0\}}$$

$$D_g^{c,c'} = \left[\phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1,2}_{\{g\}} + \left[\zeta_{g,c,c'}\right]^3_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

- A semi-functional key of type 3 also has random $\mathfrak{R}_2$ components in all secret key encodings, except that now the $\mathfrak{R}_2$ components of K_0 and K_1 are uncorrelated with the $D_g^{c,c'}$. The $\mathfrak{R}_2$ components of the $D_g^{c,c'}$ are still correlated:

$$K_0 = \left[\alpha + \theta \prod_{w \in W} \phi_w \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^1_{\{g:g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0]^{2,3}_{\{g:g \in \mathsf{C}\} \cup \{0\}}$$

$$K_1 = \left[\theta \prod_{w \in I_2} a_w^{y_w} \overline{a_w^{y_w}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + [\zeta_1]^{2,3}_{\{0\}}$$

$$D_g^{c,c'} = \left[\phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1,2}_{\{g\}} + \left[\zeta_{g,c,c'}\right]^3_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

- A semi-functional key of type 4 has random uncorrelated components in $\mathfrak{R}_2$ for all secret key encodings:

$$K_0 = \left[\alpha + \theta \prod_{w \in W} \phi_w \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^1_{\{g:g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0]^{2,3}_{\{g:g \in \mathsf{C}\} \cup \{0\}}$$

$$K_1 = \left[\theta \prod_{w \in I_2} a_w^{y_w} \overline{a_w^{y_w}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + [\zeta_1]^{2,3}_{\{0\}}$$

$$D_g^{c,c'} = \left[\phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1}_{\{g\}} + \left[\zeta_{g,c,c'}\right]^{2,3}_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

4.3 Proof Overview

At the highest level, our proof strategy will be similar to existing dual systems proofs. We prove security though a sequence of hybrids. First, we transform the challenge ciphertext to be semi-functional. Then, one-by-one, we transform the secret keys to be semi-functional (type 1). For each secret key, the transformation involves several hybrids, where we first make the secret key semi-functional (type 2), and then semi-functional (type 3), and then finally semi-functional (type 1). In previous dual systems proofs, the analog of moving from type 2 to type 3 secret keys was performed by an information theoretic argument. For our scheme, this is no longer possible. However, we will show that the transformation is still possible using a mix of computational and information theoretic arguments.

Our hybrids will be as follows:

- H_{real} is the real security experiment.
- In H_0, the challenge ciphertext is semi-functional. H_{real} and H_0 are proved indistinguishable using assumption 1.
- In H_i for $i = 1, ..., q$, the challenge ciphertext is semi-functional, secret keys $j \leq i$ are semi-functional (type 1), and secret keys $j > i$ are normal. We prove that H_{i-1} and H_i are indistinguishable using several intermediate hybrids:

– In H_i^c, the challenge ciphertext is semi-functional, secret keys $j < i$ are semi-functional (type 1), secret keys $j > i$ are normal, and secret key i is semi-functional type 2. We prove that H_{i-1} and H_i^c are indistinguishable using assumption 2.

– H_i^u, is identical to H_i^c, except that the ith secret key is semi-functional type 3. In previous dual system works, H_i^u and H_i^c are statistically indistinguishable following an information theoretic argument. However, this is not the case for our scheme. Instead, we prove the hybrids computaitonally indistinguishable using several intermediate hybrids, relying on assumption 3 and an information theoretic argument. We note that the hybrids vary depending on whether the ith secret key query occurs before of after the challenge ciphertext query.

  Finally, H_i^u is proved indistinguishable from H_i using assumption 2.

Lastly, in hybrid H_q, we show that the message encryption key is statistically hidden from the adversary, meaning the adversary has negligible advantage.

4.4 The Actual Proof

We prove adaptive security through a sequence of hybrids. The high level strategy is similar to previous dual system proofs: we first make the challenge ciphertext semi-functional, and then we gradually transform every secret key one-by-one to be semi-functional (type 1). However, the method we use to make each secret key semi-functional will be different from previous works. Once all of the secret keys are semi-functional, we can argue that the challenge message encryption key is statistically hidden from the adversary.

H_{Real}**:** This is the real security game (as defined in Section 2.1). Let q_{before} be the number of secret key queries made by the attacker before the challenge ciphertext, and q be the total number of secret key queries.

Next, we transform the challenge ciphertext to be semi-functional:

H_0**:** This is the same as H_{Real}, except that the challenge ciphertext given to the adversary is *semi-functional*, meaning its terms are encoded in $\mathfrak{R}_1 \times \mathfrak{R}_2$, rather than just in $\mathfrak{R}_1$. Specifically, the challenge ciphertext, for attribute string x, is

$$C_0^* = \left[\overline{\theta^*} \prod_{w \in W} \overline{\phi_w^*}\right]^{1,2}_{\{\overline{g}: g \in \mathsf{C}\} \cup I_1 \cup \{1\}}$$

$$C_1^* = \left[\overline{\theta^*} \prod_{w \in I_1} a_w^{x_w} \overline{a_w^{x_w}} \overline{b_{\mathsf{out}}^1}\right]^{1,2}_{I_1 \cup \{1\}}$$

$$\overline{D_g^{c,c'*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}} \overline{b^c_{\mathsf{left}(g),g}} \overline{b^{c'}_{\mathsf{right}(g),g}} \overline{a^{g(c,c')}_{\mathsf{out}(g)}}\right]^{1,2}_{\{\overline{g}\}}$$

The term H^* is then

$$\left[\alpha \overline{\theta^*} \prod_{w \in W} \overline{\phi_w^*}\right]^{1,2}_{\mathbb{U}}$$

Lemma 1. *If assumption 1 holds, then H_{Real} and H_0 are computationally indistinguishable.*

Proof. Let $\mathbb{V} = \{g : g \in \mathsf{C}\} \cup \{0\}$, $\mathbb{W}_1 = I_1 \cup \{1\}$, and $\mathbb{W}_2 = \{\overline{g} : g \in \mathsf{C}\}$. Obtain the challenge for assumption 1: $\left\{R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}}, T_i = [t_i]^{1,2}_{\{i\}}, U_i\right\}_{i\in\mathbb{U}}$ where $U_i = [u_i]^1_{\{i\}}$ or $U_i = [u_i]^{1,2}_{\{i\}}$. In addition, we also receive randomizers for each of the singleton sets, as well as the index sets $\mathbb{U}$, $\mathbb{V}$, $\mathbb{W}_1$, and $\mathbb{W}_1 \cup \mathbb{W}_2$. We will simulate the view of the adversary as follows. Set the master public key as

$$\begin{aligned} MPK = \Big(F = &\prod_{i\in\{g:g\in\mathsf{C}\}\cup\{0\}} T_i \prod_{i\in\{\overline{g}:g\in\mathsf{C}\}\cup I_1\cup\{1\}} R_i, G = \prod_{i\in\{\overline{g}:g\in\mathsf{C}\}\cup I_1\cup\{1\}} R_i, \\ &\left\{\overline{E_g^{c,c'}} = \overline{b^c_{\mathsf{left}(g),g}}\,\overline{b^{c'}_{\mathsf{right}(g),g}}\,\overline{a^{g(c,c')}_{\mathsf{out}(g)}} R_{\overline{g}}\right\}_{g\in\mathsf{C};c,c'\in\{0,1\}}, \\ &\{J_{w,c} = a^c_w \overline{a^c_w} R_w\}_{w\in I_1;c\in\{0,1\}}, B = b^1_{\mathsf{out}} R_1\Big) \end{aligned}$$

Assumption 1 gives us the randomization parameters necessary to re-randomize all of the public key components. Thus, this simulates the public key correctly, formally setting $\alpha = \prod_{i\in\{g:g\in\mathsf{C}\}\cup\{0\}} t_i$. Similarly, we can simulate secret keys using R_i, S_i, and T_i and the randomization parameters as follows: the jth secret key, for attribute $y^{(j)}$, is computed as

$$\begin{aligned} K^{(i)}_0 &= \prod_{i\in\{g:g\in\mathsf{C}\}\cup\{0\}} T_i + \theta^{(j)} \prod_{w\in W} \phi^{(j)}_w \prod_{w\in\mathsf{C}} (s_w \overline{s_w}) \prod_{i\in\{g:g\in\mathsf{C}\}\cup\{0\}} R_i + \zeta_0 \prod_{i\in\{g:g\in\mathsf{C}\}\cup\{0\}} S_i \\ K^{(i)}_1 &= \theta^{(j)} \prod_{w\in I_2} \left(a^{y^{(i)}_w}_w \overline{a^{y^{(i)}_w}_w}\right) b^1_{\mathsf{out}} R_0 + \zeta_1 S_0 \\ D^{c,c'(i)}_g &= \phi^{(j)}_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)} R_g + \zeta_{g,c,c'} S_g \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\} \end{aligned}$$

Namely, use the T_i to generate the α component in the $\mathfrak{R}_1 \times \mathfrak{R}_2$, R_i to generate the component in $\mathfrak{R}_1$, and S_i to randomize the $\mathfrak{R}_3$. Finally, we generate the challenge ciphertext as

$$\begin{aligned} C^*_0 &= \prod_{i\in\{\overline{g}:g\in\mathsf{C}\}\cup I_1\cup\{1\}} U_i \\ C^*_1 &= \prod_{w\in I_1} a^{x_w}_w \overline{a^{x_w}_w}\,\overline{b^1_{\mathsf{out}}} \prod_{i\in I_1\cup\{1\}} U_i \\ \overline{D^{c,c'*}_g} &= \overline{b^c_{\mathsf{left}(g),g}}\,\overline{b^{c'}_{\mathsf{right}(g),g}}\,\overline{a^{g(c,c')}_{\mathsf{out}(g)}} U_{\overline{g}} \\ H^* &= \prod_{i\in\{\overline{g}:g\in\mathsf{C}\}\cup I_1\cup\{1\}} U_i \prod_{i\in\{g:g\in\mathsf{C}\}\cup\{0\}} T_i \end{aligned}$$

Again, we can use the randomization parameters to re-randomize each of the ciphertext components.

If $U_i = [u_i]^1_{\{i\}}$, this amounts to setting $\overline{\phi_{\mathsf{out}(g)}} = u_{\overline{g}}$ and $\overline{\theta} = \prod_{i\in I_1\cup\{1\}} u_i$ in hybrid H_{Real}. If $U_i = [u_i]^{1,2}_{\{i\}}$, this amounts to the same settings of variables, but in hybrid H_0. Thus, any distinguisher for H_{Real} and H_0 will also break assumption 1. □

Now that the challenge ciphertext is semi-functional, we gradually turn the secret keys into semi-functional (type 1). We do this one secret key at a time:

H_i**:** For each secret key query made before the challenge ciphertext, we define the hybrid H_i which is identical to H_0 except that the first i secret keys are *semi-functional (type 1)*, which means that K_0 has a random component in $\mathfrak{R}_2$. In other words, the jth key, for $j \leq i$ and attribute $y^{(j)}$, is generated as

$$K_0^{(j)} = \left[\alpha + \theta^{(j)} \prod_{w \in W} \phi_w^{(j)} \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^1_{\{g : g \in \mathsf{C}\} \cup \{0\}} + \left[\zeta_0^{(j)}\right]^{2,3}_{\{g : g \in \mathsf{C}\} \cup \{0\}}$$

$$K_1^{(j)} = \left[\theta^{(j)} \prod_{w \in I_2} a_w^{y_w^{(j)}} \overline{a_w^{y_w^{(j)}}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + \left[\zeta_1^{(j)}\right]^3$$

$$D_g^{c,c'(j)} = \left[\phi^{(j)}_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^1_{\{g\}} + \left[\zeta^{(j)}_{g,c,c'}\right]^3_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

Now we argue the indistinguishability of hybrids H_{i-1} and H_i. We do so by defining the additional intermediate hybrids:

H_i^c**:** This is the same as hybrid H_{i-1}, except that the ith secret key is *semi-functional (type 2)*, which means that the secret key terms have random but correlated $\mathfrak{R}_2$ components for all encodings:

$$K_0^{(i)} = \left[\alpha + \theta^{(i)} \prod_{w \in W} \phi_w^{(i)} \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^{1,2}_{\{g : g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0^{(i)}]^3_{\{g : g \in \mathsf{C}\} \cup \{0\}}$$

$$K_1^{(i)} = \left[\theta^{(i)} \prod_{w \in I_2} a_w^{y_w^{(i)}} \overline{a_w^{y_w^{(i)}}} b^1_{\mathsf{out}}\right]^{1,2}_{\{0\}} + [\zeta_1^{(i)}]^3_{\{0\}}$$

$$D_g^{c,c'(i)} = \left[\phi^{(i)}_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1,2}_{\{g\}} + [\zeta^{(i)}_{g,c,c'}]^3_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

Lemma 2. *If assumption 2 holds, then H_{i-1} and H_i^c are computationally indistinguishable.*

Proof. Let $\mathbb{V} = \{g : g \in \mathsf{C}\} \cup \{0\}$, $\mathbb{W}_1 = I_1 \cup \{1\}$, and $\mathbb{W}_2 = \{\overline{g} : g \in \mathsf{C}\}$. Obtain the challenge for assumption 2: $\left\{R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}} T_i = [t_i]^{1,2}_{\{i\}}\right\}_{i \in \mathbb{U}}, U = [u]^{2,3}_{\mathbb{V}}, \{V_i\}_{i \in \mathbb{V}}$, where $V_i = [v_i]^{1,3}_{\{i\}}$ or $V_i = [v_i]^{1,2,3}_{\{i\}}$. We also receive randomizers, which, similar to the proof of Lemma 1, are sufficient for re-randomizing all of the encodings handed out during simulation. Using the R_i and T_i, we can simulate the public parameters as in the proof of Lemma 1, formally setting $\alpha = \prod_{\{g : g \in \mathsf{C}\} \cup \{0\}} t_i$. We can also generate the challenge ciphertext using the T_i as before (now the T_i are not the challenge elements, but lie in the $\mathfrak{R}_1 \times \mathfrak{R}_2$). Functional secret keys for $j > i$ will be generated as before using the R_i, S_i, and T_i. Semi-functional (type 1) secret keys for $j < i$ can also be generated using the R_i, S_i, T_i, and U (basically, randomize K_0 in both $\mathfrak{R}_2$ and $\mathfrak{R}_3$ using U instead of S_i). Finally, we generate the secret key for i as

$$K_0^{(i)} = \prod_{i \in \{g : g \in \mathsf{C}\} \cup \{0\}} T_i + \prod_{w \in \mathsf{C}} (s_w \overline{s_w}) \prod_{i \in \{g : g \in \mathsf{C}\} \cup \{0\}} V_i + \zeta_0 \prod_{i \in \{g : g \in \mathsf{C}\} \cup \{0\}} S_i$$

$$K_1^{(i)} = \prod_{w \in I_2} \left(a_w^{y_w^{(i)}} \overline{a_w^{y_w^{(i)}}} \right) b^1_{\mathsf{out}} V_0 + \zeta_1 S_0$$

$$D_g^{c,c'(i)} = b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)} V_g + \zeta_{g,c,c'} S_g \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

If $V_i = [v_i]^{1,3}$, this amounts to setting $\phi^{(i)}_{\mathsf{out}(g)} = v_g$, $\theta^{(i)} = v_0$, $\zeta_0^{(i)} = \zeta_0 + v_0 \prod_{g \in \mathsf{C}} v_g \prod_{w \in \mathsf{C}} s_w \overline{s_w}$, $\zeta_1^{(i)} = \zeta_1 + v_0 \prod_{w \in I_2} \left(a_w^{y_w^{(i)}} \overline{a_w^{y_w^{(i)}}} \right) b^1_{\mathsf{out}}$, and $\zeta^{(i)}_{g,c,c'} = \zeta_{g,c,c'} + v_g b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}$ in hybrid H_{i-1}. If $V_i = [v_i]^{1,2,3}$, this amounts to the same settings of variables, but in hybrid H_i^c. Thus, any distinguisher for H_{i-1} and H_i^c will also break assumption 2. □

Next, we start making changes to the ith secret key or challenge ciphertext based on the challenge ciphertext or ith secret key query, respectively. We will have to split into two cases: if $i \leq q_{before}$, we change the challenge ciphertext based on the ith secret key, and if $i > q_{before}$, we change the ith secret key based on the challenge ciphertext. We now give the $i \leq q_{before}$ case, and will later return to the other case.

$H_i^{c,gar}$ for $i \leq q_{before}$: This is the same as hybrid H_i^c, except that we *garble* the semi-functional challenge ciphertext. Roughly, this means the following. When using the ith secret key to decrypt the challenge ciphertext, for each gate g, only a single one of the $\overline{D_g^{c,c'*}}$ value will be used. For the other three values, we replace the component in $\mathfrak{R}_2$ with a completely random component.

More precisely, let $y^{(i)}$ be the ith secret key attribute, and x be the challenge ciphertext attribute. Let c_w be the value of wire w when evaluating $\mathsf{C}(x, y^{(i)})$. Then compute the $\overline{D_g^{c,c'*}}$ values as

$$\overline{D_g^{c_{\mathsf{left}(g)}, c_{\mathsf{right}(g)}*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}} \overline{b^{c_{\mathsf{left}(g)}}_{\mathsf{left}(g),g}} \overline{b^{c_{\mathsf{right}(g)}}_{\mathsf{right}(g),g}} \overline{a^{c_{\mathsf{out}(g)}}_{\mathsf{out}(g)}} \right]^{1,2}_{\{\overline{g}\}}$$

$$\overline{D_g^{c,c'*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}} b^c_{\mathsf{left}(g),g} \overline{b^{c'}_{\mathsf{right}(g),g}} \overline{a^{g(c,c')}_{\mathsf{out}(g)}} \right]^1_{\{\overline{g}\}} + [\gamma_{g,c,c'}]^2_{\{\overline{g}\}} \text{ for all } (c,c') \neq (c_{\mathsf{left}(g)}, c_{\mathsf{right}(g)})$$

Lemma 3. *If assumption 3 holds, then for $i \leq q_{before}$, H_i^c and $H_i^{c,gar}$ are computationally indistinguishable.*

Proof. We prove Lemma 3 by iteratively garbling each gate in the circuit in topological order, each time relying on assumption 3 to prove that the per-gate garbling is undetectable. We show the case for the first gate, the rest of the gates being similar. Assume that $c_{\mathsf{left}(g)} = c_{\mathsf{right}(g)} = 0$, the other cases being similar. Let $e = \overline{g}$, and obtain the challenge for assumption 4 for e: $\left(\left\{ R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}}, T_i = [t_i]^2_{\{i\}} \right\}_{i \in \mathbb{U}}, U = [t_{\overline{g}} u]^2_{\{\overline{g}\}}, V = [t_{\overline{g}} v]^2_{\{\overline{g}\}}, W = [t_{\overline{g}} w]^2_{\{\overline{g}\}} \right)$ where w is random in $\mathfrak{R}$ or $w = uv$. We generate all of the terms as in H_i^c except for the $\overline{D_g^{c,c'*}}$ terms, which

we generate as:

$$\overline{D_g^{0,0*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^0_{\mathsf{left}(g),g} b^0_{\mathsf{right}(g),g}}\,\overline{a^{g(0,0)}_{\mathsf{out}(g)}}\right]^1_{\{\overline{g}\}} + \overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^0_{\mathsf{left}(g),g} b^1_{\mathsf{right}(g),g}}\,\overline{a^{g(0,1)}_{\mathsf{out}(g)}} T_{\overline{g}}$$

$$\overline{D_g^{0,1*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^0_{\mathsf{left}(g),g} b^1_{\mathsf{right}(g),g}}\,\overline{a^{g(0,1)}_{\mathsf{out}(g)}}\right]^1_{\{\overline{g}\}} + \overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^0_{\mathsf{left}(g),g} b^1_{\mathsf{right}(g),g}}\,\overline{a^{g(0,1)}_{\mathsf{out}(g)}} U$$

$$\overline{D_g^{1,0*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^1_{\mathsf{left}(g),g} b^0_{\mathsf{right}(g),g}}\,\overline{a^{g(1,0)}_{\mathsf{out}(g)}}\right]^1_{\{\overline{g}\}} + \overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^1_{\mathsf{left}(g),g} b^0_{\mathsf{right}(g),g}}\,\overline{a^{g(1,0)}_{\mathsf{out}(g)}} V$$

$$\overline{D_g^{1,1*}} = \left[\overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^1_{\mathsf{left}(g),g} b^1_{\mathsf{right}(g),g}}\,\overline{a^{g(1,1)}_{\mathsf{out}(g)}}\right]^1_{\{\overline{g}\}} + \overline{\phi^*_{\mathsf{out}(g)}}\,\overline{b^1_{\mathsf{left}(g),g} b^1_{\mathsf{right}(g),g}}\,\overline{a^{g(1,1)}_{\mathsf{out}(g)}} W$$

Note that only the ith secret key and the challenge ciphertext have non-zero and correlated values in $\mathfrak{R}_2$. Therefore, in $\mathfrak{R}_2$, the values $\overline{a_w^{1-y_w^{(i)}}}$ for $w \in I_2$ and the values $\overline{a_w^{1-x_w}}$ for $w \in I_1$ are never used. This means $\overline{b_{w,g}^{1-x_w}}$ and $\overline{b_{w,g}^{1-y_w^{(i)}}}$ are random and independent of all the other terms. Moreover, these only appear in the $\overline{D_g^{c,c'*}}$ components of the challenge ciphertext. Therefore, if $w = uv$, the above formally replaces $\overline{\phi_{\mathsf{out}(g)}}$ in $\mathfrak{R}_2$ with $\overline{\phi_{\mathsf{out}(g)}} \cdot t_{\overline{g}}$, $b^1_{\mathsf{left}(g),g}$ with $b^1_{\mathsf{left}(g),g} \cdot v$ and $b^1_{\mathsf{right}(g),g}$ with $b^1_{\mathsf{right}(g),g} \cdot u$ in hybrid H_i^c. If w is random, then the $\mathfrak{R}_2$ components of $\overline{D_g^{0,1*}}$, $\overline{D_g^{1,0*}}$, and $\overline{D_g^{1,1*}}$ are each random and independent of all other terms. Finally, once the $\mathfrak{R}_2$ component of $\overline{D_g^{c,c'*}}$ for $(c,c') \neq (0,0)$ is random and independent, $\overline{\phi_g^*}$ appears only in $\overline{D_g^{0,0*}}$, so the $\mathfrak{R}_2$ component this encodings is also random and independent. Thus, in the case $w = uv$, the gate g is garbled.

Note that once g is garbled, $\overline{a^{1-g(0,0)}_{\mathsf{out}(g)}}$ is not used in $\mathfrak{R}_2$. This means that the $\overline{b^{1-g(0,0)}_{\mathsf{out}(g),g'}}$ values are random and only appear in the corresponding $\overline{D_{g'}^{c,c'*}}$ values. Thus, once we have garbled both gates feeding into an internal gate, we can garble the internal gate as well using the same strategy. Doing this for all gates completes the proof.

□

$H_i^{u,gar}$ **for** $i \leq q_{before}$**:** This is the same as hybrid $H_i^{c,gar}$, except that the ith secret key is *semi-functional (type 3)*. In other words, the $\mathfrak{R}_2$ components of K_0 and K_1 are uncorrelated. This means the ith secret key is generated as

$$K_0^{(i)} = \left[\alpha + \theta^{(i)} \prod_{w \in W} \phi_w^{(i)} \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^1_{\{g: g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0^{(i)}]^{2,3}$$

$$K_1^{(i)} = \left[\theta^{(i)} \prod_{w \in I_2} a_w^{y_w^{(i)}} \overline{a_w^{y_w^{(i)}}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + [\zeta_1^{(i)}]^{2,3}$$

$$D_g^{c,c'(i)} = \left[\phi^{(i)}_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1,2}_{\{g\}} + [\zeta^{(j)}_{g,c,c'}]^3_{\{g\}} \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

Lemma 4. *For $i \leq q_{before}$, $H_i^{c,gar}$ and $H_i^{u,gar}$ are statistically indistinguishable.*

Proof. After garbling, in $\mathfrak{R}_2$, $\overline{a^1_{\mathsf{out}}}$ is never used, and $\overline{b^0_{\mathsf{out}}}$ was never used from the start. This means the $\mathfrak{R}_2$ component $\overline{s_{\mathsf{out}}}$ is uniformly random and independent of all the other terms. Moreover, the

$\mathfrak{R}_2$ component of $\overline{s_{\mathsf{out}}}$ only appears in $K_0^{(i)}$. Thus, the $\mathfrak{R}_2$ component of $K_0^{(i)}$ is uniformly random and uncorrelated with $K_1^{(i)}$ and the $D_g^{c,c'(i)}$. □

H_i^u **for** $i \leq q_{before}$**:** This is the same as hybrid $H_i^{u,gar}$, except that the challenge cihertext is ungarbled (that is, the challenge ciphertext is semi-functional, and the ith secret key is semi-functional (type 3)).

Lemma 5. *If assumption 3 holds, then for $i \leq q_{before}$, $H_i^{u,gar}$ and H_i^u are computationally indistinguishable.*

Proof. This is proved in an analogous manner to Lemma 3 by un-garbling each gate in reverse. The only difference is that, in each intermediate garbling step, the $\mathfrak{R}_2$ components of $K_0^{(i)}$ and $K_1^{(i)}$ are independent and random. □

Lemma 6. *If assumption 2 holds, then for $i \leq q_{before}$, H_i^u and H_i are computationally indistinguishable.*

Proof. The proof is similar to the proof of Lemma 2. Let $\mathbb{V} = \{g : g \in \mathsf{C}\} \cup \{0\}$, $\mathbb{W}_1 = I_1 \cup \{1\}$, and $\mathbb{W}_2 = \{\overline{g} : g \in \mathsf{C}\}$. Next, we obtain the challenge for assumption 2 using $\mathbb{V}, \mathbb{W}_1, \mathbb{W}_2$: $\left\{R_i = [1]^1_{\{i\}}, S_i = [s_i]^3_{\{i\}} T_i = [t_i]^{1,2}_{\{i\}}\right\}_{i \in \mathbb{U}}, U = [u]^{2,3}_{\mathbb{V}}, \{V_i\}_{i \in \mathbb{V}}$ be the challenge, where $V_i = [v_i]^{1,3}_{\{i\}}$ or $V_i = [v_i]^{1,2,3}_{\{i\}}$. Using the R_i, we can simulate the public parameters as before, formally setting $\alpha = \prod_{i \in \{g : g \in \mathsf{C}\} \cup \{0\}} t_i$. We can also generate the challenge ciphertext using the T_i similar to before. Secret keys for $j \neq i$ will be generated as in the proof of Lemma 2 using the R_i, S_i, and T_i and U. Finally, we generate the ith secret key as

$$K_0^{(i)} = \prod_{i \in \{g : g \in \mathsf{C}\} \cup \{0\}} T_i + \theta \prod_{w \in W} \phi_w \prod_{w \in \mathsf{C}} s_w \overline{s_w} \prod_{i \in \{g : g \in \mathsf{C}\} \cup \{0\}} V_i + \zeta_0 U$$

$$K_1^{(i)} = \theta \prod_{w \in I_2} a_w^{y_w^{(i)}} \overline{a_w^{y_w^{(i)}}} b^1_{\mathsf{out}} V_0$$

$$D_g^{c,c'(i)} = \phi_{\mathsf{out}(g)} b^c_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)} V_g + \zeta_{g,c,c'} S_g \text{ for } g \in \mathsf{C}; c, c' \in \{0,1\}$$

If $V_i = [v_i]^{1,2,3}_{\{i\}}$, then this amounts to setting $\theta^{(i)} = v_0 \theta$, $\phi^{(i)}_{\mathsf{out}(g)} = v_g \phi_{\mathsf{out}(g)}$, and independent random values for the $\zeta^{(i)}$ terms in hybrid $H_i^{u,gar}$. If $V_i = [v_i]^{1,3}_{\{i\}}$, then this amounts to the same setting of variables[4] in hybrid H_i^{gar}. Thus, any distinguisher for hybrid $H_i^{u,gar}$ and H_i^{gar} successfully breaks assumption 2. □

Now we have addressed all the secret key queries up to the challenge ciphertext, and made them uncorrelated semi-functional keys. Next, we need to do the same for the keys made after the challenge ciphertext. However, the above sequence of hybrids will not work to move from H_i^c to H_i. In particular, in hybrid $H_i^{c,gar}$, we garbled the challenge ciphertext based on the ith secret key query, which meant we needed the ith query to occur before the challenge. We can no longer rely on this, and we must therefore adjust the hybrids accordingly. Basically, we perform the same garbling steps on the secret keys themselves, based on the challenge ciphertext (which now occurs before the ith query).

[4] the randomness values change, but remain independent

$H_i^{c,gar}$ **for** $i > q_{before}$**:** This is the same as H_i^c, except that we garble the ith secret key (as opposed to garbling the challenge ciphertext). Roughly, this means the following. When using the ith secret key to decrypt the challenge ciphertext, for each gate g, only a single one of the $D_g^{c,c'(i)}$ values will be used. For the other three values, we replace the $\mathfrak{R}_2$ component with random.

More precisely, let $y^{(i)}$ be the ith secret key attribute, and x be the challenge ciphertext attribute. Let c_w be the value of wire w when evaluating $\mathsf{C}(x, y^{(i)})$. Then compute the $D_g^{c,c'(i)}$ values as

$$D_g^{c_{\mathsf{left}(g)},c_{\mathsf{right}(g)}(i)} = \left[\phi^{(i)}_{\mathsf{out}(g)} b^{c_{\mathsf{left}(g)}}_{\mathsf{left}(g),g} b^{c_{\mathsf{right}(g)}}_{\mathsf{right}(g),g} a^{c_{\mathsf{out}(g)}}_{\mathsf{out}(g)}\right]^{1,2}_{\{g\}} + [\zeta^{(i)}_{g,c,c'}]^3_{\{g\}}$$

$$D_g^{c,c'(i)} = \left[\phi^{(i)}_{\mathsf{out}(g)} b^{c}_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1}_{\{g\}} + [\zeta^{(i)}_{g,c,c'}]^{2,3}_{\{g\}} \text{ for all } (c,c') \neq (c_{\mathsf{left}(g)}, c_{\mathsf{right}(g)})$$

Lemma 7. *If assumption 3 holds, then for $i > q_{before}$, H_i^c and $H_i^{c,gar}$ are computationally indistinguishable.*

Proof. This is exactly analogous to Lemma 3 and proved in a similar manner. □

$H_i^{u,gar}$ **for** $i > q_{before}$**:** This is the same as hybrid $H_i^{c,gar}$, except that the ith secret key is *semi-functional (type 4)*, where the $\mathfrak{R}_2$ components of every term of the ith secret key are uncorrelated (as apposed to type 3 keys, where the $D_g^{c,c'}$ terms remain correlated with each other). In other words, the secret key is generated as

$$K_0^{(i)} = \left[\alpha + \theta^{(i)} \prod_{w \in W} \phi_w^{(i)} \prod_{w \in \mathsf{C}} s_w \overline{s_w}\right]^1_{\{g : g \in \mathsf{C}\} \cup \{0\}} + [\zeta_0^{(i)}]^{2,3}$$

$$K_1^{(i)} = \left[\theta^{(i)} \prod_{w \in I_2} a_w^{y_w^{(i)}} \overline{a_w^{y_w^{(i)}}} b^1_{\mathsf{out}}\right]^1_{\{0\}} + [\zeta_1^{(i)}]^{2,3}$$

$$D_g^{c,c'(i)} = \left[\phi^{(i)}_{\mathsf{out}(g)} b^{c}_{\mathsf{left}(g),g} b^{c'}_{\mathsf{right}(g),g} a^{g(c,c')}_{\mathsf{out}(g)}\right]^{1}_{\{g\}} + [\zeta^{(i)}_{g,c,c'}]^{2,3}_{\{g\}}$$

Lemma 8. *For $i > q_{before}$, $H_i^{c,gar}$ and $H_i^{u,gar}$ are statistically indistinguishable.*

Proof. After garbling, in $\mathfrak{R}_2$, a^1_{out} is never used, and b^0_{out} was never used from the start. This means the $\mathfrak{R}_2$ component s_{out} is uniformly random and independent of all the other terms. Moreover, the $\mathfrak{R}_2$ component of s_{out} only appears in $K_0^{(i)}$. Thus, the $\mathfrak{R}_2$ component of $K_0^{(i)}$ is uniformly random and uncorrelated with $K_1^{(i)}$ and the $D_g^{c,c'(i)}$. At this point, in $\mathfrak{R}_2$, the $\phi_g^{(i)}$ only appears in $D_g^{c_{\mathsf{left}(g)},c_{\mathsf{right}(g)}(i)}$, and therefore the $\mathfrak{R}_2$ component of $D_g^{c_{\mathsf{left}(g)},c_{\mathsf{right}(g)}(i)}$ becomes independent of all the other terms. The $\mathfrak{R}_2$ components of the other $D_g^{c,c'(i)}$ is already independent as a result of garbling. □

H_i^u **for** $i > q_{before}$**:** This is the same as hybrid $H_i^{u,gar}$, except that the ith secret key is semi-functional (type 3), meaning that the $\mathfrak{R}_2$ components of $K_0^{(i)}$ and $K_1^{(i)}$ are uncorrelated, but the components if $D_g^{c,c'}$ are correlated.

Lemma 9. *If assumption 3 holds, then for $i > q_{before}$, $H_i^{u,gar}$ and H_i^u are computationally indistinguishable.*

Proof. This is proved analogously to Lemma 5, except that we un-garble each gate of the secret key in reverse, while keeping the $\mathfrak{R}_2$ components of $K_0^{(i)}$ and $K_1^{(i)}$ uncorrelated. □

Lemma 10. *If assumption 2 holds, then for $i > q_{before}$, H_i^u and H_i are computationally indistinguishable.*

Proof. The proof is identical to the proof of Lemma 6. □

At this point, we have made all of the secret keys semi-functional. Now, we claim that the message encryption key is statistically close to random:

Lemma 11. *In hybrid H_q, the adversary has negligible advantage*

Proof. Notice that now the $\mathfrak{R}_2$ component of α only appears in H, which is hidden from the adversary except through the message encryption key $K_{enc} = \mathsf{ext}(\mathsf{params}, H)$. Therefore, by the extraction property of the graded encoding, the value K_{enc} is statistically close to a uniform random bit string in $\{0,1\}^\lambda$. □

References

[BGK+14] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 221–238, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany.

[BR14] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 1–25, San Diego, CA, USA, February 24–26, 2014. Springer, Berlin, Germany.

[BS02] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. Cryptology ePrint Archive, Report 2002/080, 2002. `http://eprint.iacr.org/2002/080`.

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Germany.

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17, Athens, Greece, May 26–30, 2013. Springer, Berlin, Germany.

[GGH+13b] 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 Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press.

[GGH+13c] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 479–499, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Germany.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GLSW14] Craig Gentry, Allison Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. Cryptology ePrint Archive, Report 2014/309, 2014. `http://eprint.iacr.org/2014/309`.

[GLW14] Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 426–443, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany.

[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, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 89–98, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press. Available as Cryptology ePrint Archive Report 2006/309.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 545–554, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[LOS+10] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 191–208, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany.

[PST14] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 500–517, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany.

[PT82] Christos H. Papadimitriou and John N. Tsitsiklis. On the complexity of designing distributed protocols. *Information and Control*, 53(3):211–218, 1982.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

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

[Wat14] Brent Waters. A punctured programming approach to adaptively secure functional encryption. Cryptology ePrint Archive, Report 2014/588, 2014. `http://eprint.iacr.org/`.