Adaptively Secure Broadcast Encryption with Small System Parameters

Mark Zhandry
Stanford University
mzhandry.stanford.edu

Abstract

We build the first public-key broadcast encryption system that simultaneously achieves adaptive security against arbitrary number of colluders, has small system parameters, and has a security proof based on non-interactive falsifiable assumptions. Our scheme is built from composite order multilinear maps and enjoys a ciphertext overhead, private key size, and public key size that are are all poly-logarithmic in the total number of users. Previous broadcast schemes with similar parameters are either proven secure in a weaker static model, or rely on powerful tools such as program obfuscation and involve non-falsifiable assumptions.

1 Introduction

A broadcast encryption (BE) scheme [FN93] consists of N users, each with their own secret key, and a broadcaster. The broadcaster can dynamically choose any set S of users, and broadcast an encryption of a message so that each user in S can decrypt the broadcast with their own secret key, but users outside of S cannot, even if they all collude and pool their secret keys. Broadcast encryption has applications to paid digital TV and radio services, where broadcasts are encrypted to the current set of subscribers. Broadcast encryption also has applications to access control in encrypted file systems and more generally to group communication. In this work, we will be considering *public key* broadcast schemes, where anyone can play the role of broadcaster and encrypt.

Broadcast encryption admits a trivial solution, where each user's secret key is the secret decryption key for a public key encryption scheme, and the broadcast key consists of all the corresponding public keys. To encrypt to a set S, the broadcaster encrypts separately to each public key corresponding to users in S. The scheme can easily be proved adaptively secure, based on the semantic security of the underlying public key encryption scheme.

Therefore, the main interest in broadcast encryption is to minimize the parameter sizes. In the trivial system, public keys have size proportional to the number of users N, and secret keys are constant size. Since the ciphertext size must at a minimum encode the entire message m and recipient set S[1], the measure of interest for ciphertexts is the *overhead*, namely the amount of information that must be transmitted in addition to (the description of) S and the symmetric encryption of the actual plaintext. In the trivial system, such overhead can be made proportional to the size $|S|$ of the recipient set.

[1] With the secret key sk_i for user i, it is possible to determine if $i \in S$ by running the decryption procedure and seeing if it succeeds. With all secret keys, it is therefore possible to completely reconstruct S.

Boneh, Gentry, and Waters [BGW05] give the first broadcast scheme with sub-linear sized ciphertexts from bilinear maps, called the BGW construction. This scheme has constant size ciphertexts and secret keys (in terms of the number of users N), but a public key that is still linear in N. One trade-off of their scheme is that it is only proved secure in a *static model*, where the adversary may choose the set S the challenge ciphertext is encrypted to, but must commit to S before even seeing the public parameters. Some other schemes based on bilinear maps [DPP07, GW09] have been proven *adaptively secure*, where the adversary may choose S adaptively after seeing several secret keys. However, the public broadcast key in all of these schemes is at least linear in the maximum number of recipients.

Existing constructions based on multilinear maps. A t-linear map consists of t groups $\mathbb{G}_1, \ldots, \mathbb{G}_t$, along with generators $g_1, \ldots, g_t$, and a bilinear pairing procedure e where $e(g_i^a, g_j^b) = g_{i+j}^{ab}$. The subscripts on $\mathbb{G}_1, g_1$ are usually omitted, and the group $\mathbb{G} = \mathbb{G}_1$ is called the source group. $\mathbb{G}_t$ is called the target group, and the other groups $\mathbb{G}_2, \ldots, \mathbb{G}_{t-1}$ are called intermediate groups or intermediate levels. We will call g_i^a a "level i encoding" of a.

The first constructions of broadcast encryption from multilinear maps [BS02, GGH13a, CLT13, BW13] where all *secret key* schemes, where the broadcast key has to be kept secret. Moreover, for N users, these schemes require N-linear maps. In all current constructions of N-linear maps [GGH13a, CLT13, LSS14, GGH14], the description of the map as well as group elements have size at least $\omega(N)$. This leads to broadcast schemes with super-linear secret keys. The schemes above can be converted into public key schemes, but at the cost of including a group element in the ciphertext. This would then give a scheme with super-linear ciphertexts as well, worse than the trivial scheme. Therefore, in order for multilinear maps to give interesting broadcast encryption in the public key setting, we require the level n of multilinearity to be much less that the number of users N.

Boneh, Waters, and Zhandry [BWZ14] show how to build broadcast encryption for N users from $O(\log N)$-linear maps by generalizing the BGW construction above. In their scheme, all parameters — ciphertext overhead, secret key size, and public key size — are poly-logarithmic in N. By increasing the number of users to 2^λ, they are even able to obtain an identity-based scheme with constant-size parameters (namely, with no upper bound on the size of the recipient set). However, similar to the BGW scheme, their scheme is only proved statically secure. Boneh, Waters, and Zhandry additionally present a scheme derived from [GW09]. However, they are unable to prove security relative to static assumptions, and instead prove security in a generic model for multilinear maps, obtaining adaptive security in this model.

Constructions based on obfuscation. Boneh and Zhandry [BZ14] show how to build broadcast encryption from indistinguishability obfuscation (iO) [BGI+01, GGH+13b], where the ciphertext size and secret key size are independent of the number of users. Their scheme also has the novel property of being distributed, where each user chooses their own secret key. However, the public keys in their scheme consist of obfuscated programs whose size is at least $\Omega(N)$, and actually much larger with current obfuscation implementations [GGH+13b, AGIS14].

Ananth et al. [ABG+13] show how to shrink the public key size in Boneh and Zhandry's scheme [BZ14], but at the cost of losing the distributed property, and security only being proved in the static model. Their scheme relies on a strengthening of obfuscation, called differing inputs obfuscation (diO) [BGI+01, BCP14, ABG+13]. Zhandry [Zha14] shows how to resurrect the distributed property and achieve adaptive security using a primitive called witness PRFs, which can

be seen as a special case of obfuscation. However, Zhandry requires a strong extractable notion of security for witness PRFs. Both diO and extractable witness PRFs are non-falsifiable assumptions. Moreover, Gentry et al. [GGHW14] give evidence that the most general forms of these assumptions may not hold, and avoiding the attack in [GGHW14] often requires the use of application-specific assumptions. The weaker variants of diO and extractable witness PRFs (namely iO and standard security for witness PRFs) *can* be obtained from static falsifiable assumptions [GLW14, GLSW14], but the security proof involves complexity leveraging.

On complexity leveraging. For many cryptographic protocols, such as digital signatures, attribute-based encryption, and functional encryption, it is possible to achieve adaptive security from static security though complexity leveraging. Essentially, the reduction guesses the adversary's challenge before seeing the public key, and will abort if the adversary's challenge does not match the guess. This approach requires assuming the sub-exponential hardness of the underlying cryptographic assumption.

In the case of broadcast encryption, adaptive security can be obtained from static security in this way by guessing the challenge set S before seeing the public parameters, with a 2^{-N} chance of guessing correctly where N is the number of users. Therefore, the scheme must not be polynomial-time breakable in the static setting, except with probability significantly smaller than 2^{-N}. However, such security level necessarily involves setting the security parameter λ to be larger than N. Since parameters grow at least linearly with λ, this results in secret keys, ciphertexts, and public keys all being at least linear in the number of users, worse than the trivial system.

By combining multiple components, it may be possible to use complexity leveraging on some components and not others and still obtain adaptive security. Boneh and Zhandry [BZ14] do exactly this, where the only components that require complexity leveraging appear in the public key. Thus they are able to obtain semi-statically secure broadcast encryption with constant-sized ciphertexts and secret keys. However, whenever a component of a broadcast scheme uses a reduction that loses a 2^N factor, *some* parameter must grow at least linearly in the number of users.

1.1 Our Contributions

In this work, we give an adaptively secure broadcast scheme where all parameters — ciphertexts, secret keys, and public keys — are poly-logarithmic in the number of users. Our scheme is proved secure using polynomial reductions to simple natural assumptions on composite-order symmetric multilinear maps. The scheme is based on the generically secure scheme of Boneh, Waters, and Zhandry [BWZ14] (henceforth called the BWZ scheme), which in turn is based on the Gentry and Waters [GW09] scheme (the GW scheme). Interestingly, while the GW scheme was proved secure, the proof does not carry over to the BWZ scheme because of additional correlations between public key components. Instead, as mentioned above, Boneh, Waters, and Zhandry prove the scheme secure in a generic model of multilinear maps.

We further modify the BWZ scheme, and prove full adaptive security while preserving the poly-logarithmic sizes for ciphertexts, secret keys, and public keys of the BWZ scheme. Our proof follows the dual system framework [Wat09, Wee14], and security is based on subgroup decision-style assumptions, similar to those recently made by Garg et al. [GGHZ14], as well as a multilinear DDH-style assumption. We give a comparison of our work to existing works in Table 1.

Similar to [BWZ14], we can set the number of users to be 2^λ for a security parameter λ, and obtain an identity-based scheme that is both adaptively secure, and allows for an unbounded number

of recipients without affecting ciphertext size.

Table 1: Comparing parameters sizes and security of our scheme to some existing protocols. N is the maximum number of users, m is the maximum number of receivers (for schemes where $m < N$ is determined at setup time). $|bk|, |ct|, |sk|$ respectively denote the size of the broadcast key, ciphertext overhead, and each user's secret key. "RO" denotes that the security proof is in the random oracle model, and "NF" represents that the underlying assumptions are non-falsifiable. The column labeled "type" indicates which schemes are identity-based (id), public broadcast key (pk), or secret broadcast key (sk). Finally, the "class" column indicates which schemes are based on public key encryption (PKE), bilinear maps (BM), multilinear maps (MLM), or obfuscation-related primitives (Obf).

Scheme	$\|bk\|$	$\|ct\|$	$\|sk\|$	security	type	class
Trivial	$O(N)$	$O(\|S\|)$	$O(1)$	adaptive	pk	PKE
[BGW05]	$O(N)$	$O(1)$	$O(1)$	static	pk	BM
	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(1)$			
[Del07]	$O(m)$	$O(1)$	$O(1)$	static	id	BM
[GW09]	$O(m)$	$O(1)$	$O(1)$	semi-static	pk	BM
	$O(m)$	$O(1)$	$O(1)$	adapt. (RO)	id	
	$O(\sqrt{m})$	$O(\sqrt{\|S\|})$	$O(1)$	adaptive		
([BS02] or [BW13]) and ([GGH13a] or [CLT13])	$N^{O(1)}$	0	$N^{O(1)}$	static	sk	MLM
[ABG+13], [Zha14]	$(\log N)^{O(1)}$	$(\log N)^{O(1)}$	$(\log N)^{O(1)}$	adapt. (NF)	pk	Obf
[BWZ14]	$(\log N)^{O(1)}$	$(\log N)^{O(1)}$	$(\log N)^{O(1)}$	static	pk	MLM
	$O(1)$	$O(1)$	$O(1)$		id	
This work	$(\log N)^{O(1)}$	$(\log N)^{O(1)}$	$(\log N)^{O(1)}$	adaptive	pk	MLM
	$O(1)$	$O(1)$	$O(1)$		id	

1.2 Technical Difficulties

In the dual system framework, the challenge ciphertext and each secret key the adversary receives are gradually altered into a *semi-functional* form, where semi-functional secret keys cannot reveal any information about a semi-functional ciphertext, but otherwise decryption always works as expected (in other words, a semi-functional key can decrypt a normal ciphertext, and a normal secret key can decrypt a semi-functional ciphertext). Once the ciphertext and secret keys are semi-functional, security becomes information theoretic.

A crucial step in much of the dual system literature [Wat09, LOS+10, LW10, Wee14] is an information theoretic step for each secret key. In this step, a secret key is altered, and the change is information-theoretically undetectable exactly because the secret key is not allowed to decrypt

the challenge ciphertext. In other words, if the adversary had a secret key that could decrypt the challenge, this step would be detectable. It is exactly because this step is information theoretic that the dual system schemes obtain adaptive security.

In the case of broadcast encryption, this step provably *cannot* be information theoretic while maintaining small ciphertexts. The reason is that the number of recipient sets is 2^N, while the ciphertexts space has size $2^{|ct|} = 2^{o(N)}$. For large N, there will necessarily be multiple sets S giving the same ciphertext. Therefore, security of low-overhead broadcast schemes must involve some form of collision resistance, and this need for collision resistance breaks the information theoretic step.

In more detail, suppose an adversary randomly chooses a set S, asks for all the secret keys outside of S, and then challenges on S. He should not be able to decrypt the resulting challenge ciphertext ct. However, there is some other set S' such that ct is an encryption to S'. With probability at least $1/2$, S' will not be a subset of S, and will therefore contain a user $i \notin S$ for which the adversary has a secret key. Therefore, the adversary *is* allowed to decrypt ct, in the sense that he could decrypt if he could determine S'. If the information theoretic step were valid, it would mean that changing the secret key for user i would be undetectable to even computationally unbounded adversaries. But such an adversary could interpret ct as an encryption to S', which he *can* decrypt (because he has the secret key for user $i \in S'$), and therefore this change would be detectable. The only apparent way to resolve this contradiction is to rely on computational assumptions for this step.

Relying on computational assumptions presents several challenges for our scheme. First, the assumption we need is a multilinear analog of the DDH assumption. However, the assumption needs to hold in mid-levels of the multilinear map (since all components of the scheme exist in intermediate levels), which is usually not the case in the symmetric setting. The analog in the bilinear setting is that the DDH problem — distinguishing (g, g^a, g^b, g^{ab}) from (g, g^a, g^b, g^c) for random a, b, c — is easy in bilinear groups. This is because we can always "lift" the challenge g^c to $e(g, g)^c$ by pairing with g, which we can then compare with $e(g^a, g^b) = e(g, g)^{ab}$,

The natural workaround is to use asymmetric multilinear maps, which severely restricts the operations the adversary can perform, and thus allows more decisional problems to be hard, analogous to how DDH can hold in asymmetric bilinear groups. However, the asymmetry also restricts the operations that can be performed by our *reduction*. As a result, the proofs for the other computational steps become more challenging. In particular, we will need to embed the challenge elements in multiple levels. Because of the limited interaction between levels in the asymmetric setting, we actually need a separate challenge element in every level. Moreover, we would need to embed the challenge elements in a way so that the elements in different levels have correlations consistent with the scheme. Unfortunately, it does not appear that the reduction can create these correlations on its own, and instead the assumption itself must provide correlated challenge elements. This results in somewhat complicated assumptions.

Another possibility is to continue to use symmetric multilinear maps, but try to modify the scheme in such a way that the computational assumption needed *does* hold for mid-levels of symmetric multilinear maps. For example, in the bilinear setting, consider the distributions $(g, g^a, g^b, g^c, g^{abc})$ and (g, g^a, g^b, g^c, g^d) for random a, b, c, d. Given (g, g^a, g^b, g^c), it is not possible to even compute $e(g, g)^{abc}$, meaning even though we can "lift" our challenge g^d to the target group by pairing with g, it does not help us in distinguishing the two distributions. Basically, by having the number of variables multiplied in the exponent exceed the levels of multilinearity, we arrive at an assumption that presumably holds.

This is the general approach we follow. However, this direction presents its own set of difficulties

during the security reduction. In particular, the naive way of implementing the above idea is to have ciphertexts and secret keys consisting of group elements encoded at level $i < t$, but whose exponents are the product of $n > t$ elements, and these different elements are highly correlated with each other. During simulation, if we are given uncorrelated challenges, it becomes difficult to reproduce this correlation because we can only multiply at most i elements together to arrive at our ciphertext or secret key elements, but some how need to simulate correlations between $n \geq i + 2$ variables. Therefore, we have two competing interests: we need to restrict operations an adversary can perform while maximizing the operations our reduction can perform. We show how to achieve a balance between these two by carefully controlling exactly which levels of the map elements are given out at. The result is that the level of multilinearity we need is somewhat larger than in the BWZ scheme [BWZ14]. However, we argue that the *effective* multilinearity, in terms of the affect of multilinearity on parameter sizes, is essentially the same as in the BGW scheme.

The second perhaps more interesting challenge is that maintaining adaptivity while transitioning to a static computational assumption is problematic. In the information theoretic setting, adaptivity is free. However, in the computational setting, the transformation step needs to be handled carefully. In particular, the step involves simulating the challenge ciphertext and a secret key, but the simulations occur at different points in the reduction, and we have to start simulating with incomplete information. For example, if the ciphertext query occurs before the secret key query, we will have to simulate the ciphertext before knowing which user the secret key will be for. Somehow we need to embed our *static* assumption into the ciphertext while reserving the ability to generate any subsequent secret key query the adversary may ask for (namely, for users outside the challenge set). Conversely, if the ciphertext query occurs after the secret key query, we will have to embed our assumption into the secret key, and ensure that we can simulate all possible ciphertexts the adversary may ask for.

This means the reduction will have to embed the challenge differently, depending on whether the challenge ciphertext or the particular secret key query we are modifying come first. Similar difficulties were faced (and overcome) by Attrapadung [Att14] in constructing adaptively secure functional encryption for regular languages, and by Garg et al. [GGHZ14] in constructing adaptively secure attribute-based encryption for circuits. In spite of these difficulties, we show how to perform this computational step using a static assumption, thus preserving adaptivity.

2 Preliminaries

2.1 Broadcast Encryption

We begin by defining broadcast encryption. A (public key) identity-based broadcast encryption scheme consists of four randomized algorithms:

$\mathsf{Setup}(\mathcal{ID})$: Sets up a broadcast scheme for identity space $\mathcal{ID}$. It outputs public parameters params as well as a master secret key msk

$\mathsf{KeyGen}(\mathsf{msk}, u)$: Takes the master secret key and a user $u \in \mathcal{ID}$ and outputs a secret key sk_u for user u.

$\mathsf{Enc}(\mathsf{params}, S)$: The encryption algorithm takes the public parameters and a polynomial sized set $S \subseteq \mathcal{ID}$ of recipients, and produces a pair (Hdr, K). We refer to Hdr as the header, and K as the message encryption key.

The message is encrypted using a symmetric encryption scheme with the key K to obtain a ciphertext c. The overall ciphertext is (Hdr, c).

$\mathsf{Dec}(\mathsf{params}, u, \mathsf{sk}_u, S, \mathsf{Hdr})$: The decryption algorithm takes the header Hdr and the secret key for user u, and if $u \in S$, outputs the message encryption key K. If $u \notin S$, the decryption algorithm outputs $\perp$.

To actually decrypt the overall ciphertext (Hdr, c), user u runs Dec to obtain K, and then decryption c using K to obtain the message.

For correctness, we require that the decryption algorithm always succeeds when it is supposed to. That is, for every $(\mathsf{params}, \mathsf{msk})$ output by $\mathsf{Setup}(\mathcal{ID})$, every set $S \subseteq \mathcal{ID}$, every sk_u output by $\mathsf{KeyGen}(\mathsf{msk}, u)$, and (Hdr, K) outputted by $\mathsf{Enc}(\mathsf{params}, S)$ where $u \in S$, that $\mathsf{Dec}(\mathsf{params}, u, \mathsf{sk}_u, S, \mathsf{Hdr}) = K$.

If we set $\mathcal{ID} = [N]$ for a polynomial N, and redefine the setup procedure to also run $\mathsf{KeyGen}(\mathsf{msk}, u)$ for all $i \in [N]$, then we can eliminate the need for a persistent master secret key. We then recover the notion of (non-identity-based) broadcast encryption [FN93].

We now define adaptive security using the following experiment $\mathtt{EXP}(b)$ on adversary $\mathcal{A}$:

Setup: The challenger runs $(\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{Setup}(\mathcal{ID})$, and gives $\mathcal{A}$ the public key params.

Secret Key Queries: $\mathcal{A}$ may adaptively make secret key queries for user u. In response, the challenger runs $\mathsf{sk}_u \leftarrow \mathsf{KeyGen}(\mathsf{msk}, u)$ and gives sk_u to $\mathcal{A}$.

Challenge: $\mathcal{A}$ submits a set $S^* \subset \mathcal{ID}$, subject to the restriction that $u \notin S^*$ for any user u requested in a secret key query. The challenger lets $(\mathsf{Hdr}^*, K_0^*) \leftarrow \mathsf{Enc}(\mathsf{params}, S^*)$. If $b = 0$, the challenger gives (Hdr^*, K_0^*) to the adversary. If $b = 1$, the challenger computes a random key K_1^* and gives (Hdr^*, K_1^*) to the adversary.

More Secret Key Queries: $\mathcal{A}$ may continue making secret key queries for users $u \notin S^*$

Guess: $\mathcal{A}$ produces a guess b' for b.

Using a simple hybrid argument, we can assume the adversary makes only a single challenge query. Let W_b be the event that $\mathcal{A}$ outputs 1 in $\mathtt{EXP}(b)$. We define the adaptive advantage of $\mathcal{A}$, as

$$\mathtt{BE}_{\mathcal{A}}^{(\mathrm{adv})} = |\Pr[\mathtt{W_0}] - \Pr[\mathtt{W_1}]|$$

Definition 2.1. A broadcast encryption scheme is adaptively secure if, for all polynomial time adversaries $\mathcal{A}$, $\mathtt{BE}_{\mathcal{A}}^{(\mathrm{adv})}$ is negligible.

2.2 Multilinear Maps

Now, we describe multilinear maps, mostly following [GGH13a, CLT13, GLW14]. We will use graded encoding notation, rather than group notation. Our notation for composite order maps comes from [GGHZ14], except that we will use symmetric maps instead of asymmetric maps.

A symmetric multilinear map consists of two algorithms:

$\mathsf{Setup}(t, \mathfrak{R}, k)$: Sets up an t-linear symmetric map over the ring $\mathfrak{R}$, where $\mathfrak{R} = \mathfrak{R}_1 \times \cdots \times \mathfrak{R}_k$ for some hidden rings $\mathfrak{R}_i$ of roughly equal size. It outputs public parameters Params, and a secret key sk.

$\mathsf{Encode}(\mathsf{sk}, \alpha \in \mathfrak{R}, i \leq t)$: The secret encoding procedure[2] outputs the "encoding" of α at some level $i \leq t$. Levels above t are not permitted. We write the output as $[\alpha]_i$.

$+, -$: There are two binary public procedures $+$ and $-$ written in infix notation. $+$ takes as input two encodings at the same level, and outputs an encoding of the sum. That is, $[\alpha]_i + [\beta]_i = [\alpha + \beta]_i$. The level i must be the same on both summands. The operation $-$ simply negates the second summand: $[\alpha]_i - [\beta]_i = [\alpha]_i + [-\beta]^i$. For any i, the encodings $[\alpha]_i$ form a commutative group under $+, -$. We call the set of encodings at level 1 the *source group*, and each level-1 encoding is a *source group encoding* or *source group element*. We call the set of encodings at level t the *target group*, and each encoding a *target group encoding* or *target group element*.

$\times$: We will also represent this by $\cdot$. $\times$ takes as input two encodings $[\alpha]_i$ and $[\beta]_j$. We require that $i + j \leq t$. It outputs the encoding $[\alpha\beta]_{i_j}$. We will also associate ring elements $\alpha \in \mathfrak{R}$ with "level-0 encodings" $\alpha \equiv [\alpha]_0$. Thus we can compute $\alpha \times [\beta]_i = [\alpha]_0 \times [\beta]_i = [\alpha\beta]_i$.

$\mathsf{ext}(\mathsf{Params}, e)$: The public extraction procedure takes a level t encoding e, and extracts a λ-bit string s from e. We require that, for any $i \in [k]$, that if α is sampled uniformly at random from $\mathfrak{R}_i$, then $\mathsf{ext}(\mathsf{Params}, [\alpha]_t)$ is statistically close to a uniform string, even given Params.

In our applications, a master party will know the rings $\mathfrak{R}_i$, and can therefore project any $\alpha \in \mathfrak{R}$ down to a sub-product of the $\mathfrak{R}_i$. Let $\mathfrak{R}_T = \prod_{i \in [T]} \mathfrak{R}_i$. The master party will also coincide with the secret key holder. We will therefore define the combined secret project-and-encode procedure $\mathsf{Encode}(\mathsf{sk}, \alpha, i, T)$ which first projects α to $\mathfrak{R}_T$ obtaining α_T, and then encodes α_T at level i. For convenience, we will write such encodings as $[\alpha]_i^T$: for example, if $T = \{1, 3\}$, the we write $[\alpha]_i^{1,3}$. We note that encodings of elements in subrings obey the following properties, which can be derived from the definitions above:

$$[\alpha]_i^T + [\beta]_i^{T'} = [\alpha + \beta]_i^{T \cap T'} + [\alpha]_i^{T \setminus T'} + [\beta]_i^{T' \setminus T} \qquad [\alpha]_i^T \times [\beta]_{i'}^{T'} = [\alpha\beta]_{i+i'}^{T \cap T'}$$

To instantiate multilinear maps, we can use Gentry et al.'s variant [GLW14] of the Coron-Lepoint-Tibouchi (CLT) multilinear maps [CLT13]. This variant is designed to emulate multilinear groups of composite order, and to allow assumptions regarding subgroups of the multilinear groups[3]. However, current candidate multilinear maps, including the CLT maps, do not quite fit the abstraction presented in Section 2, which complicate applications of the abstraction. However, these complications are easily overcome in our application.

- Encodings are not unique. That is, there are multiple valid encodings of each α relative to each set S, and $\mathsf{Encode}(\mathsf{Params}, \alpha, i)$ is a randomized procedure, which samples from the set of possible encodings of α relative to S. In this case, the notation $[\alpha + \beta]_i = [\alpha]_i + [\beta]_i$ no longer makes sense. Instead, we require that the sum of an encoding of α and an encoding of β is an encoding of $\alpha + \beta$ relative to the same set. A similar statement holds for multiplying encodings.

[2] Current multilinear map candidates [GGH13a, CLT13] allow either a secret or public encoding procedure. The public version of the procedure requires publishing slightly more information in Params, which may impact the security of the maps. Our scheme does not require a public encoding procedure, so we use the secret procedure to maximize security. However, our scheme is still correct, and our assumptions still presumably hold, even when using the public encoding procedure.

[3] The Garg-Gentry-Halevi multilinear maps [GGH13a] do not satisfy these subgroup assumptions.

- The fact that encodings are not unique means that an encoding may reveal the sequence of operations that lead to that element. Therefore, we require a re-randomization procedure to re-randomize the encoding, and make it statistically independent of the procedures that lead to that element. Note that when performing operations, we do not need to re-randomize after every operation; instead, we will only need to re-randomize when we send the encoding to someone else. All current graded encoding candidates support this randomization procedure.

- For functionality, we will also need that $\mathsf{ext}(\mathsf{Params}, e) = \mathsf{ext}(\mathsf{Params}, e')$ for any two encodings e, e' of the same ring element. This is true of current candidates.

- Encodings have noise. This means, after every operation, the noise grows, and if the noise grows too large, ext may fail to give the correct output. Re-randomization also increases the noise. Note that additions only cause mild noise growth, so the noise growth is dominated by the number of multiplications and the re-randomization procedure. In our applications, we will only re-randomize a constant number of times, and the number of multiplications is bounded by multilinearity of the map, so it is straightforward to set the parameters so that the noise never grows too large.

- The public interface does not give direct access to the ring $\mathfrak{R}$ itself. The schemes do allow users to sample "level 0" encodings of random elements $[\alpha]_0$, which can then be lifted to higher levels by multiplying by a level 1 encoding of 1, $[1]_1$, included in the public parameters. For our scheme, this functionality will be sufficient.

- Finally, in CLT encodings, we do not have complete control over the rings $\mathfrak{R}_i$. In particular, $\mathfrak{R}_i = \mathbb{Z}_{N_i}$ for some composite integers N_i with large prime factors. However, for simplicity we describe our application in terms of $\mathbb{Z}_{p_i}$ for prime p_i, making $\mathfrak{R}_i$ a field. Note that in $\mathbb{Z}_{N_i}$, the set of zero divisors is sparse, and the prime factors are unknown, meaning a randomly chosen element will be invertible. Therefore, the rings $\mathbb{Z}_{N_i}$ are in some sense "as good as" a field, and will suffice for our purposes.

We mostly describe our scheme in the ideal multilinear map setting, rather than relying on the particular CLT candidate. We do this for two reasons:

- To cleanly present our ideas without the complications involved in non-ideal multilinear maps. However, to give a more complete picture using current candidates, we describe at a high level how to cope with the difficulties above as they arise.

- To make our results more general. If new candidate multilinear maps are found that side-step the issues above or have new issues of their own, then having our scheme described generically facilitates porting our scheme over to the new map.

3 Our Construction

We now give our construction from composite-order symmetric multilinear maps. The construction is based on the third and final construction of Boneh, Waters, and Zhandry [BWZ14] (the BWZ scheme), which in turn is based on the broadcast scheme of Gentry and Waters [GW09] (the GW scheme). While the GW scheme could be proven secure in a *semi-static* model, it contained a large public key. Boneh, Waters, and Zhandry showed how to shrink the public key using multilinear

maps, but where unable to prove security of the resulting BWZ scheme relative to non-interactive assumptions. They proved that their scheme had no trivial attacks by proving it adaptively secure, but in a generic model for multilinear maps.

Construction 3.1. $\mathsf{Setup}(\ell)$: takes as input the length ℓ of identities. That is, $\mathcal{ID} = \{0,1\}^\ell$. Choose three large primes p_1, p_2, p_3, let $\mathfrak{R}_i = \mathbb{Z}_{p_i}$, and let $\mathfrak{R} = \mathfrak{R}_1 \times \mathfrak{R}_2 \times \mathfrak{R}_3$. Run the setup algorithm for an multilinear map, $\mathsf{Setup}'(\ell(\ell+2), \mathfrak{R}, 3)$, to construct an $t = \ell(\ell+2)$-linear map with three subgroups and parameters Params[4]. Draw a random $\alpha, \gamma, \delta \in \mathfrak{R}$ and for $i \in [\ell]$ and $b \in \{0,1\}$, draw random $\beta_{i,b} \in \mathfrak{R}$. The public key is

$$\mathsf{pk} = \left(\mathsf{Params}, \left\{A_{i,b} = [\beta_{i,b}]^{1,2}_{\ell+1}\right\}_{i\in[\ell],b\in\{0,1\}}, B = [\gamma]^1_\ell, L = [\delta]^1_{\ell(\ell+1)}, I = [\alpha\gamma]^1_{\ell(\ell+2)}\right)$$

The master secret key for the system is $\mathsf{msk} = (\alpha, \gamma, \delta, \{p_i\}_{i\in[3]}, \{\beta_{i,b}\}_{i\in[n],b\in\{0,1\}})$

$\mathsf{Enc}(\mathsf{pk}, S)$: Choose a random $t \in \mathfrak{R}$ and compute

$$J = t \cdot I = [\alpha\gamma t]^1_{\ell(\ell+2)} \quad , \quad C = t \cdot B = [t\gamma]^1_\ell \quad ,$$
$$D = t \cdot \left(L + \sum_{\mathbf{v}\in S} \prod_{i\in[\ell]} A_{i,b}\right) = \left[t\left(\delta + \sum_{\mathbf{v}\in S} \prod_{i\in[\ell]} \beta_{i,v_i}\right)\right]^1_{\ell(\ell+1)}$$

The message encryption key is $K = \mathsf{ext}(\mathsf{Params}, J)$ and the ciphertext header is $\mathsf{Hdr} = (C, D)$

$\mathsf{KeyGen}(\mathsf{msk}, \mathbf{u})$: Pick a random $r^{\mathbf{u}} \in \mathfrak{R}$ and random $s_i^{\mathbf{u}} \in \mathfrak{R}$ for $i \in [\ell]$. Also pick random $\eta_{i,b}, \zeta_i, \xi, \mu \in \mathfrak{R}$. Output the secret key components

$$\mathsf{sk}^{\mathbf{u}} = \Bigg(\left\{E^{\mathbf{u}}_{i,b} = [s_i^{\mathbf{u}}\beta_{i,b}]^1_{\ell+1} + [\eta_{i,b}]^3_{\ell+1}\right\}_{i\in[\ell],b\in\{0,1\}},$$
$$\left\{F^{\mathbf{u}}_i = [r^{\mathbf{u}} s_i^{\mathbf{u}} \beta_{i,1-u_i}]^1_{\ell+1} + [\zeta_i]^3_{\ell+1}\right\}_{i\in[\ell]}, G^{\mathbf{u}} = \left[\gamma r^{\mathbf{u}} \prod_{i\in[\ell]} s_i^{\mathbf{u}}\right]^1_\ell + [\xi]^3_\ell,$$
$$H^{\mathbf{u}} = [\alpha]^{1,2}_{\ell(\ell+1)} + \left[r^{\mathbf{u}} \prod_{i\in[\ell]} s_i^{\mathbf{u}} \left(\delta + \prod_{i\in[\ell]} \beta_{i,u_i}\right)\right]^1_{\ell(\ell+1)} + [\mu]^3_{\ell(\ell+1)}\Bigg)$$

Notice that all the $\mathfrak{R}_3$ components are just uniformly random, and, with the exception of α in $H^{\mathbf{u}}$, all the $\mathfrak{R}_2$ components are empty. Note that from $\mathsf{sk}^{\mathbf{u}}$, it is possible to compute

$$Z^{\mathbf{u}}_{\mathbf{v}} = \left[r^{\mathbf{u}} \prod_{i\in[\ell]} s_i^{\mathbf{u}} \prod_{i\in[\ell]} \beta_{i,v_i}\right]^1_{\ell(\ell+1)} + [\phi^{\mathbf{u}}_{\mathbf{v}}]^3_{\ell(\ell+1)}$$

[4] In CLT encodings, the p_i must be composite integers with large prime factors. However, as described in Section 2, such p_i are suffice for our application.

for any $\mathbf{v} \neq \mathbf{u}$ where $\phi_{\mathbf{v}}^{\mathbf{u}}$ is some ring element as follows. Choose some i^* such that $\mathbf{u} \neq \mathbf{v}$. Since $\mathfrak{R}_2$ is empty, and $\mathfrak{R}_3$ is arbitrary, we can just focus on $\mathfrak{R}_1$. $F_{i^*}^{\mathbf{u}}$ encodes $r^{\mathbf{u}} s_{i^*} \beta_{i^*,1-u_{i^*}} = r^{\mathbf{u}} s_{i^*} \beta_{i^*,v_{i^*}}$. Therefore, compute

$$F_{i^*}^{\mathbf{u}} \cdot \prod_{i \in [\ell] \setminus \{i^*\}} E_{i,v_i}^{\mathbf{u}} = \left[r^{\mathbf{u}} s_{i^*}^{\mathbf{u}} \beta_{i,v_i} \prod_{i \neq i^*} (s_i^{\mathbf{u}} \beta_{i,v_i}) \right]_{\ell(\ell+1)} = \left[r^{\mathbf{u}} \prod_{i \in [\ell]} s_i^{\mathbf{u}} \prod_{i \in [n]} \beta_{i,v_i} \right]_{\ell(\ell+1)} \quad \text{(in } \mathfrak{R}_1\text{)}$$

$\mathsf{Dec}(\mathsf{sk}^{\mathbf{u}}, (C, D))$: Compute

$$J' = \left(H^{\mathbf{u}} + \sum_{\mathbf{v} \in S \setminus \{\mathbf{u}\}} Z_{\mathbf{v}}^{\mathbf{u}} \right) \cdot C - D \cdot G$$

and then let $K' = \mathsf{ext}(\mathsf{Params}, J')$

Correctness. It suffices to show $J' = J$ in decryption. Notice that since C and D have no $\mathfrak{R}_2$ or $\mathfrak{R}_3$ component, the result I' will only have a $\mathfrak{R}_1$ component. Therefore, we have

$$\begin{aligned} J' = \Bigg[& \left(\alpha + r^{\mathbf{u}} \prod_{i \in [\ell]} s_i^{\mathbf{u}} \left(\delta + \prod_{i \in [n]} \beta_{i,u_i} \right) + r^{\mathbf{u}} \prod_{i \in [\ell]} s_i^{\mathbf{u}} \sum_{\mathbf{v} \in S \setminus \{\mathbf{u}\}} \prod_{i \in [\ell]} \beta_{i,v_i} \right) \gamma t \\ & - t \left(\delta + \sum_{\mathbf{v} \in S} \prod_{i \in [n]} \beta_{i,v_i} \right) \gamma r^{\mathbf{u}} \prod_{i \in [\ell]} s_i^{\mathbf{u}} \Bigg]_{\ell(\ell+2)}^{1} \\ = & \left[\alpha \gamma t \right]_{\ell(\ell+2)}^{1} = J \end{aligned}$$

Implementation Details. Since we instantiate our multilinear maps using the non-ideal noisy maps of [CLT13], we need to ensure that all components that might be given to the adversary are re-randomized. This means we need to publish re-randomization parameters for the levels $\ell, \ell(\ell+1)$. We will also need to keep in the master secret key randomization parameters for level $\ell + 1$. Notice that the total number of sets that require re-randomization parameters is just 3, meaning the total size of all parameters (ciphertext, master secret, public, and user secret keys) are polynomial in ℓ, or equivalently logarithmic in the number of users.

We will also need to ensure our scheme can handle the noise growth associated with the encryption and decryption operations (plus the operations in the security reduction). The total number of additions and multiplications that lead to the term J (from which the message encryption key is derived) is proportional to $|S|$ additions and ℓ multiplications. Since additions do not increase the noise by much, it is straight-forward but tedious to set the parameters of the scheme so that these additions can be handled. The size of the parameters will be $\mathsf{poly}(\log |S|, \ell) = \mathsf{polylog}(|\mathcal{ID}|)$.

In the next section, we will prove the security of our scheme.

4 Security Proof

4.1 Assumptions

We now introduce the assumptions we will be using to prove the security of our scheme. We believe our assumptions are new to this work; however they are very similar to existing assumptions

on multilinear maps. The first two assumptions are basically symmetric variants of those made by Garg et al. [GGHZ14], and can be seen as generalizing the assumptions made in dual system works [Wat09, Wee14] to the multilinear setting. Our third assumption is closely related to the multilinear DDH assumption.

First, we state our assumptions in their simplest form. Later, we will define minor variants of these assumptions that we actually use to prove security of our scheme. We can prove security relative to either set of assumptions.

Definition 4.1 (Assumption 1)**.** For any t, for a t-linear map, no efficient adversary can distinguish the following two distributions, where m, n, o, q are sampled uniformly from $\mathfrak{R}$:

$$M = [m]_1^1, N = [n]_1^{1,2}, O = [o]_1^3, Q = [q]_1^1 \text{ and}$$
$$M = [m]_1^1, N = [n]_1^{1,2}, O = [o]_1^3, Q = [q]_1^{1,2}$$

The problem of distinguishing these two cases appears hard because there is no way to isolate the $\mathfrak{R}_2$ component of Q by multiplying with M, N, O.

Definition 4.2 (Assumption 2)**.** For any ℓ, t, for a t-linear map, no efficient adversary can distinguish the following two distributions, where m, n, o, q, p are sampled uniformly from $\mathfrak{R}$:

$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_1^3, P = [p]_{t-\ell}^{2,3}, Q = [q]_1^{1,3} \text{ and}$$
$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_1^3, P = [p]_{t-\ell}^{2,3}, Q = [q]_1^{1,2,3}$$

These two appear indistinguishable as well. In order to distinguish, both the $\mathfrak{R}_1$ and $\mathfrak{R}_3$ component of Q must be eliminated, while preserving $\mathfrak{R}_2$. $\mathfrak{R}_1$ can be eliminated by multiplying with O or P, but multiplying by O eliminates $\mathfrak{R}_2$ as well. Multiplying by P does leave $\mathfrak{R}_2$, but the result is encoded at level $t - \ell + 1$. The only way to eliminate $\mathfrak{R}_3$ while preserving $\mathfrak{R}_2$ is to multiply by N, but as N is encoded at level ℓ, and we only have $\ell - 1$ levels remaining, it is not possible to carry out this step. Therefore, there appears no way to distinguish the two cases.

Definition 4.3 (Assumption 3)**.** For any t, for a t-linear map, no efficient adversary can distinguish the following two distributions:

$$M = [1]_1^1, N = [1]_1^2, O = [1]_1^3, \{P_i = [p_i]_1^2\}_{i\in[t+1]}, Q = [q]_1^2 \text{ and}$$
$$M = [1]_1^1, N = [1]_1^2, O = [1]_1^3, \{P_i = [p_i]_1^2\}_{i\in[t+1]}, Q = \left[\prod_{i\in[t+1]} p_i\right]_1^2$$

Since Q encodes the product of $t + 1$ variables, namely more than the level of multilinearity, the adversary cannot compute the product of the P_i at *any* level, so this assumption seems hard.

Derived assumptions. Here we describe two alternate assumptions for our assumptions, which are implied by the assumptions above. These assumptions are the ones we will actually use in our proof. We present these assumptions for two reasons:

- They are closer to what we will actually use in our proof, and therefore make the proof simpler.

- Our scheme *nominally* requires a $\ell(\ell+2)$-linear map. However, in current candidates [GGH13a, CLT13], what matters for setting the parameter sizes is not the multilinearity, but the number of multiplications the scheme needs to support. Therefore, the *effective* multilinearity might actually be lower. In our scheme, the number of multiplications is $O(\ell)$. However, to maintain security, we also need to incorporate the multiplications performed by our reductions. Using Assumptions 1, 2, and 3, the number of multiplications would be $\ell(\ell+2)$, meaning the effective multilinearity is still $O(\ell^2)$. However, using Assumptions 1', 2', and 3' below, the total number of multiplications would remain $O(\ell)$. Thus, by basing security on Assumptions 1', 2', and 3', we can set the effective multilinearity to $O(\ell)$. This results in a more efficient scheme.

Definition 4.4 (Assumption 1')**.** For any ℓ, t, for a t-linear map, no efficient adversary can distinguish the following two distributions, where m, n, o, q are sampled uniformly from $\mathfrak{R}$:

$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_\ell^3, O' = [o']_{\ell+1}^3, Q = [q]_1^1 \text{ and}$$
$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_\ell^3, O' = [o']_{\ell+1}^3, Q = [q]_1^{1,2}$$

The only differences between Assumption 1 and 1' are the levels N and O are encoded at, plus the addition of O'.

Lemma 4.5. *Assumption 1 implies Assumption 1'*

Proof. Suppose an adversary distinguishes the two cases in Assumption 1'. Given a challenge (M, N, O, Q) for Assumption 1, draw random $n_1, o_1, o_2 \in \mathfrak{R}$, and give the adversary the challenge $(M, n_1 N^\ell, o_1 O^\ell, o_2 O^{\ell+1}, Q)$. It is straightforward to see that the challenge the adversary sees consists of encodings of random elements at the right levels and in the correct sub-rings, consistent with Assumption 1'. Therefore, if the adversary breaks Assumption 1', it will also break Assumption 1. □

Definition 4.6 (Assumption 2')**.** For any ℓ, t, for a t-linear map, no efficient adversary can distinguish the following two distributions, where m, n, o, o', q, p are sampled uniformly from $\mathfrak{R}$:

$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_\ell^3, O' = [o']_{\ell+1}^3, P = [p]_{t-\ell}^{2,3}, Q = [q]_1^{1,3} \text{ and}$$
$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_\ell^3, O' = [o']_{\ell+1}^3, P = [p]_{t-\ell}^{2,3}, Q = [q]_1^{1,2,3}$$

The only difference between Assumption 2 and 2' is the level for O and the addition of O'.

Lemma 4.7. *Assumption 2 implies Assumption 2'*

Proof. Suppose an adversary distinguishes the two cases in Assumption 2'. Given a challenge (M, N, O, P, Q) for Assumption 1, draw random $o_1, o_2 \in \mathfrak{R}$, and give the adversary the challenge $(M, N, o_1 O^\ell, o_2 O^{\ell+1}, P, Q)$. It is straightforward to see that the challenge the adversary sees consists of encodings of random elements at the right levels and in the correct sub-rings, consistent with Assumption 2'. Therefore, if the adversary breaks Assumption 2', it will also break Assumption 2. □

Definition 4.8 (Assumption 3')**.** For any ℓ, t, for a t-linear map, no efficient adversary can

distinguish the following two distributions, where $j = \lceil (t+1)/(\ell+1) \rceil$:

$$M = [1]^1_\ell, M' = [1]^1_{\ell+1}, N = [1]^2_\ell, N' = [1]^2_{\ell+1}, O = [1]^3_\ell, O' = [1]^3_{\ell+1},$$
$$\{P_i = [p_i]^2_{\ell+1}\}_{i\in[j]}, Q = [q]^2_{(\ell+1)(j-1)} \text{ and}$$

$$M = [1]^1_\ell, M' = [1]^1_{\ell+1}, N = [1]^2_\ell, N' = [1]^2_{\ell+1}, O = [1]^3_\ell, O' = [1]^3_{\ell+1},$$
$$\{P_i = [p_i]^2_{\ell+1}\}_{i\in[j]}, Q = \left[\prod_{i\in[j]} p_i\right]^2_{(\ell+1)(j-1)}$$

Lemma 4.9. *Assumption 3 implies Assumption 3'*

Proof. Suppose an adversary distinguishes the two cases in Assumption 3'. Suppose we are given a challenge $(M, N, O, \{P_i\}_{i\in[t+1]}, Q)$ for Assumption 3, where $P_i = [p_i]^2_1$ and $Q = [\prod_{i\in[t+1]} p_i]^2_1$ or Q is a random encoding in $\mathfrak{R}_2$. Let $k = j(\ell+1)$. Notice that since $j = \lceil (t+1)/(\ell+1) \rceil$, $k \geq t+1$. For $i \in [t+2, k]$, choose random $p_i \in \mathfrak{R}$, and set $P_i = p_i N = [p_i]^2_1$. Output the tuple

$$\left(M^\ell, M^{\ell+1}, N^\ell, N^{\ell+1}, O^\ell, O^{\ell+1}, \left\{ \prod_{i'\in[\ell+1]} P_{(i-1)(\ell+1)+i'} \right\}_{i\in[j]}, \left(\prod_{i\in[t+2,k]} p_i \right) Q N^{k-\ell-2} \right)$$

The first 6 elements are encodings of 1 in $\mathfrak{R}_1, \mathfrak{R}_2, \mathfrak{R}_3$ at levels $\ell, \ell+1$, as desired. The $\prod_{i'\in[\ell+1]} P_{(i-1)(\ell+1)+i'}$ are level $\ell+1$ encodings of random independent elements p'_i. Moreover, if $Q = [\prod_{i\in[t+1]} p_i]^2_1$, then the final element is a level $k-\ell-1 = (j-1)(\ell+1)$ encoding of $\prod_{i\in[j+1]} p'_i$, and if q is random, the final element is a random encoding. Thus, we've simulated the challenge for Assumption 3, as desired. □

Because Assumptions 1', 2', and 3' follow from Assumptions 1, 2, and 3, respectively, we can base the security of our scheme on either set of assumptions. However, the reductions from the unprimed to primed assumptions involve $O(\ell)$ multiplications, which will give a total of $O(\ell^2)$ multiplications for the total reduction from our scheme to Assumptions 1, 2, and 3. We note that if we wish to set the effective multilinearity to $O(\ell)$, we would not be able to handle both the multiplications in the reductions above and those in our broadcast scheme, and therefore Assumptions 1', 2', and 3' no longer follow from Assumptions 1, 2, and 3. Therefore, we have a trade-off: a less efficient scheme from very simple assumptions, or a more efficient scheme from more complicated assumptions.

Details for non-ideal encodings. We use a multilinear map with $t = \ell(\ell+2)$. Since we will actually be using non-ideal noisy encodings, we need to publish all of the necessary parameters the simulator may need. It suffices to provide randomization parameters in the levels $1, \ell, \ell+1, \ell(\ell+1) = t-\ell$. As the total number of randomization parameters is only 4, the size of our challenges is polynomial in ℓ, or equivalently logarithmic in the number of users. We also need to make sure the parameters are set so that the noise introduced by the reduction does not cause any errors. However, our reductions perform very few operations in addition to the operations performed by our scheme, so it is easy to handle the minor extra noise growth.

4.2 Security Theorem and Proof

Here we present the main theorem of this work and the proof.

Theorem 4.10. *If Assumptions 1', 2', and 3' hold, then our scheme is adaptively secure.*

We prove security through a sequence of hybrid games, ultimately arriving at a game where information-theoretic security holds. Our hybrids will roughly follow the dual system framework of Waters [Wat09, Wee14], gradually changing the challenge ciphertext and secret keys to a *semi-functional* form. We note one important difference: in [Wat09, Wee14], there is a crucial information theoretic step that no longer holds in our setting. Instead, we need to replace the information-theoretic step with a computational one. However, this adds some complication, as this step will depend on whether the challenge ciphertext comes before or after the particular secret key we are transforming.

Hybrid *Real*. This is the normal game where, on challenge set S^*, the adversary receives the header $\mathsf{Hdr} = (C^*, D^*)$, and either the correct message encryption key K^*, or a randomly chosen key. Let ϵ be advantage the adversary has in guessing which key he is given.

Hybrid 0. This is identical to **Hybrid** *Real*, except that the challenge ciphertext (C^*, D^*) and message encryption key K^* are *semi-functional*. This means that the ciphertext is generated as:

$$J^* = [\alpha\gamma t]^{1,2}_{\ell(\ell+2)} \quad , \quad C^* = [\gamma t]^{1,2}_{\ell} \quad , \quad D^* = \left[t \left(\delta + \sum_{\mathbf{v} \in S^*} \prod_{i \in [\ell]} \beta_{i,v_i} \right) \right]^{1,2}_{\ell(\ell+1)}$$

and $K^* = \mathsf{ext}(\mathsf{Params}, J^*)$. The following is proved in Section 4.2.1:

Lemma 4.11. *Given Assumption 1',* **Hybrid** *Real and* **Hybrid 0** *are indistinguishable.*

Hybrid k**.** Hybrid k is identical to **Hybrid 0** with the challenge ciphertext being semi-functional, but the first k secret keys are also semi-functional, while the remaining secret keys are generated normally. A semi-functional key is generated as $\mathsf{sk}^{\mathbf{u}} = (\left\{E^{\mathbf{u}}_{i,b}\right\}_{i\in[b],b\in\{0,1\}}, \{F^{\mathbf{u}}_i\}_{i\in[\ell]}, G^{\mathbf{u}}, H^{\mathbf{u}})$ where:

$$E^{\mathbf{u}}_{i,b} = [s^{\mathbf{u}}_i \beta_{i,b}]^1_{\ell+1} + [\eta_{i,b}]^3_{\ell+1} \qquad F^{\mathbf{u}}_i = [r^{\mathbf{u}} s^{\mathbf{u}}_i \beta_{i,1-u_i}]^1_{\ell+1} + [\zeta_i]^3_{\ell+1}$$

$$G^{\mathbf{u}} = \left[\gamma r^{\mathbf{u}} \prod_{i\in[\ell]} s^{\mathbf{u}}_i \right]^1_{\ell} + [\xi]^3_{\ell} \qquad H^{\mathbf{u}} = \left[\alpha + r^{\mathbf{u}} \prod_{i\in[\ell]} s^{\mathbf{u}}_i \prod_{i\in[\ell]} \beta_{i,u_i} \right]^1_{\ell(\ell+1)} + [\mu]^{2,3}_{\ell(\ell+1)}$$

The only difference between a normal secret key and a semi-functional secret key is the $\mathfrak{R}_2$ component of $H^{\mathbf{u}}$, which is random for semi-functional secret keys but encodes α in normal secret keys. Also note that $k = 0$ corresponds to **Hybrid 0**, and that the only difference between **Hybrid** $k-1$ and **Hybrid** k is that secret key k goes from normal to semi-functional.

Lemma 4.12. *Given Assumptions 2' and 3',* **Hybrid** $k-1$ *and* **Hybrid** k *are indistinguishable.*

Proving 4.12 is non-trivial, and involves introducing additional intermediate hybrids:

Hybrid $k.1$**.** This is identical to **Hybrid** $k-1$, except that the kth secret key (which was normal in **Hybrid** $k-1$) is now *correlated semi-functional*. This means that it is generated as

$$E^{\mathbf{u}}_{i,b} = [s^{\mathbf{u}}_i \beta_{i,b}]^{1,2}_{\ell+1} + [\eta_{i,b}]^3_{\ell+1} \qquad F^{\mathbf{u}}_i = [r^{\mathbf{u}} s^{\mathbf{u}}_i \beta_{i,1-u_i}]^{1,2}_{\ell+1} + [\zeta_i]^3_{\ell+1}$$

$$G^{\mathbf{u}} = \left[\gamma r^{\mathbf{u}} \prod_{i\in[\ell]} s^{\mathbf{u}}_i\right]^{1,2}_{\ell} + [\xi]^3_{\ell} \qquad H^{\mathbf{u}} = \left[\alpha + r^{\mathbf{u}} \prod_{i\in[\ell]} s^{\mathbf{u}}_i \prod_{i\in[\ell]} \beta_{i,u_i}\right]^{1,2}_{\ell(\ell+1)} + [\mu]^3_{\ell(\ell+1)}$$

The only different from **Hybrid** $k-1$ is that all the secret key components are now encoded in $\mathfrak{R}_2$. The following lemma is proved in Section 4.2.2:

Lemma 4.13. *Given Assumption 2',* **Hybrid** $k-1$ *and* **Hybrid** $k.1$ *are indistinguishable.*

Hybrid $k.2$**.** This is identical to **Hybrid** $k.1$, except that the ℓth secret key is now *uncorrelated semi-functional*, which means that it is generated as

$$E^{\mathbf{u}}_{i,b} = [s^{\mathbf{u}}_i \beta_{i,b}]^{1,2}_{\ell+1} + [\eta_{i,b}]^3_{\ell+1} \qquad F^{\mathbf{u}}_i = [r^{\mathbf{u}} s^{\mathbf{u}}_i \beta_{i,1-u_i}]^{1,2}_{\ell+1} + [\zeta_i]^3_{\ell+1}$$

$$G^{\mathbf{u}} = \left[\gamma r^{\mathbf{u}} \prod_{i\in[\ell]} s^{\mathbf{u}}_i\right]^{1,2}_{\ell} + [\xi]^3_{\ell} \qquad H^{\mathbf{u}} = \left[\alpha + r^{\mathbf{u}} \prod_{i\in[\ell]} s^{\mathbf{u}}_i \left(\delta + \prod_{i\in[\ell]} \beta_{i,u_i}\right)\right]^{1}_{\ell(\ell+1)} + [\mu]^{2,3}_{[n]}$$

The only difference from **Hybrid** $k.1$ is that the $\mathfrak{R}_2$ component of $H^{\mathbf{u}}$ is random and uncorrelated with the other $\mathfrak{R}_2$ components.

Lemma 4.14. *Given Assumption 3',* **Hybrid** $k.1$ *and* **Hybrid** $k.2$ *are indistinguishable.*

In [Wat09, Wee14], this step is an information theoretic step based on the fact that user $\mathbf{u}$ is not a part of the recipient set S^*. However, in our case, the hybrids are information-theoretically distinguishable, and instead we must rely on computational arguments. However, this is problematic in the adaptive setting, because we will not necessarily know how to embed the Assumption 3' challenge until we know both the challenge set S^* and the kth identity $\mathbf{u}$. This means the reduction will depend on whether the kth secret key comes before or after the ciphertext. In Section 4.2.3, we show how to handle both cases, each with a single invocation of Assumption 3'.

To finish the proof of Lemma 4.12, we need to prove the following, proved in Section 4.2.2:

Lemma 4.15. *Given Assumption 2',* **Hybrid** $\ell.2$ *and* **Hybrid** ℓ *are indistinguishable.*

At this point, we have shown that **Hybrid** *Real* is computationally indistinguishable from **Hybrid** q, where all secret keys are semi-functional. It remains to show that the adversary has negligible advantage in **Hybrid** q:

Lemma 4.16. *Any (potentially unbounded) adversary has negligible advantage in* **Hybrid** q*.*

Proof. In **Hybrid** q, the $\mathfrak{R}_2$ component of α only appears in J^*, and is thus only visible to the adversary through $K^* = \mathsf{ext}(\mathsf{Params}, J^*)$. Therefore, the second component of J^* is random and independent of the rest of the view of the adversary, and by the extraction property of the multilinear map, K^* is statistically close to a uniform bit string in $\{0,1\}^\lambda$. □

This completes the proof of Theorem 4.10. Now we fill in the proofs of the hybrids.

4.2.1 Proof of Lemma 4.11

Proof. Obtain the challenge for Assumption 1': $M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_\ell^3, O' = [o']_{\ell+1}^3, Q$ where $Q = [q]_1^1$ or $Q_= [q]_1^{1,2}$. Also choose random $\beta'_{i,b}, \alpha', \gamma', \delta', \in \mathfrak{R}$. Let $A_{i,b} = \beta'_{i,b} \cdot M \cdot N$. This formally sets $\beta_{i,b} = mn\beta'_{i,b}$. Set $B = \gamma' M^\ell$, $I = \alpha'\gamma' M^\ell \cdot N^{\ell+1}$, and $L = \delta' M^\ell N^\ell$. This formally sets $\alpha = n^{\ell+1}\alpha'$, $\gamma = m^\ell \gamma'$, and $\delta = n^\ell m^\ell \delta'$. Note that all the formal variables are statistically close to random[5].

Next, for a secret key for user $\mathbf{u}$, choose random $s_i^{\mathbf{u}} \in \mathfrak{R}, r^{\mathbf{u}}$, as well as $\eta'_{i,b}, \zeta'_i, \xi', \mu' \in \mathfrak{R}$, and set

$$E_{i,b}^{\mathbf{u}} = s_i^{\mathbf{u}} \cdot A_{i,b} + \eta'_{i,b} O' \qquad F_i^{\mathbf{u}} = r^{\mathbf{u}} s_i^{\mathbf{u}} A_{i,b} + \zeta'_i O'$$

$$G^{\mathbf{u}} = r^{\mathbf{u}} \left(\prod_{i \in [\ell]} s_i^{\mathbf{u}} \right) \cdot B + \xi' O \qquad H^{\mathbf{u}} = \alpha' N^{\ell+1} + r^{\mathbf{u}} \left(\prod_{i \in [\ell]} s_i^{\mathbf{u}} \right) \left(L + \prod_{i \in [\ell]} A_{i,u_i} \right) + \mu' O'^\ell$$

This is consistent with the setting of variables for the public parameters, and the $\mathfrak{R}_3$ components of each term are fresh random elements. Finally, choose a random $t' \in \mathfrak{R}$ and set the challenge ciphertext elements as:

$$J^* = t'\alpha'\gamma' \cdot N^{\ell+1} \cdot Q^\ell \ , \ C^* = t'\gamma' Q^\ell \ , \ D^* = t' \left(\delta' + \sum_{\mathbf{v} \in S^*} \prod_{i \in [\ell]} \beta'_{i,v_i} \right) \cdot Q^\ell N^\ell$$

In the case where Q is encoded in $\mathfrak{R}_1$, this correctly simulates the challenge ciphertext in **Hybrid** *Real*, formally setting $t = t'q^\ell/m^\ell$. In the case where Q is encoded in $\mathfrak{R}_2$, this correctly simulates the challenge ciphertext in **Hybrid 0**, also formally setting $t = t'q^\ell/m^\ell$. Thus, if an adversary can distinguish **Hybrid** *Real* from **Hybrid 0**, our reduction will distinguish the two cases of Assumption 1', a contradiction. □

4.2.2 Proof of Lemma 4.13 and Lemma 4.15

Proof of Lemma 4.13. Obtain the challenge for Assumption 2':

$$M = [m]_1^1, N = [n]_\ell^{1,2}, O = [o]_\ell^3, O' = [o']_{\ell+1}^3, P = [p]_{\ell(\ell+1)}^{2,3}, Q$$

where $Q = [q]_1^{1,3}$ or $Q = [q]_1^{1,2,3}$. We simulate the public parameters and challenge ciphertext exactly as we did in the proof of Lemma 4.11, except that we use N instead of Q to simulate the challenge ciphertext. This formally sets $\beta_{i,b} = mn\beta'_{i,b}, \gamma = m^\ell \gamma', \alpha = n^{\ell+1}\alpha', \delta = n^\ell m^\ell \delta'$. To simulate jth secret key query, there are three cases: $j < \ell$, $j = \ell$, and $j > \ell$. Secret keys for $j > \ell$ are also simulated exactly as in the proof of Lemma 4.11. Secret keys for $j < \ell$ are simulated as:

$$E_{i,b}^{\mathbf{u}} = s_i^{\mathbf{u}} \cdot A_{i,b} + \eta'_{i,b} O' \qquad F_i^{\mathbf{u}} = r^{\mathbf{u}} s_i^{\mathbf{u}} A_{i,b} + \zeta'_i O'$$

$$G^{\mathbf{u}} = r^{\mathbf{u}} \left(\prod_{i \in [\ell]} s_i^{\mathbf{u}} \right) \cdot B + \xi' O \qquad H^{\mathbf{u}} = \alpha' N^{\ell+1} + r^{\mathbf{u}} \left(\prod_{i \in [\ell]} s_i^{\mathbf{u}} \right) \left(L + \prod_{i \in [\ell]} A_{i,u_i} \right) + \mu' P$$

[5]The variables are statistically close to random provided the set zero divisors is sparse. This holds whether $\mathfrak{R}_i$ are prime order or composite with large prime factors.

It is straightforward to see that this is the correct simulation. Secret key $j = \ell$ is simulated using the challenge element Q as

$$E^{\mathbf{u}}_{i,b} = \beta'_{i,b} s'_i \cdot N \cdot Q + \eta'_{i,b} O' \qquad F^{\mathbf{u}}_i = r^{\mathbf{u}} \beta'_{i,1-u_i} s'_i N \cdot Q + \zeta'_i O'$$

$$G^{\mathbf{u}} = r^{\mathbf{u}} \gamma' \prod_{i \in [\ell]} s'_i Q^\ell + \xi' O \qquad H^{\mathbf{u}} = \alpha' N^{\ell+1} + r^{\mathbf{u}} \prod_{i \in [\ell]} s'_i \left(\delta' + \prod_{i \in [\ell]} \beta'_{i,u_i} \right) \cdot Q^\ell N^\ell + \mu' O'^\ell$$

Notice that the O, O' components ensure that the $\mathfrak{R}_3$ component is completely random. In the case where Q is encoded relative to $\mathfrak{R}_1 \times \mathfrak{R}_3$, this correctly simulates **Hybrid** $k-1$, formally setting $s^{\mathbf{u}}_i = s'_i q/m$. In the case where Q is encoded relative to $\mathfrak{R} = \mathfrak{R}_1 \times \mathfrak{R}_2 \times \mathfrak{R}_3$, this correctly simulates **Hybrid** $k.1$ with $s^{\mathbf{u}}_i = s'_i q/m$. Thus if an adversary can distinguish the two hybrids, it will break Assumption 2, a contradiction. □

Proof of Lemma 4.15. The proof is almost identical to the proof of Lemma 4.13, except that we randomize the $\mathfrak{R}_2$ component of the $H^{\mathbf{u}}$ for the kth secret key by using P instead of O'^ℓ. □

4.2.3 Proof of Lemma 4.14

We first handle the simpler case where the kth secret key query comes before the challenge ciphertext. Let q_{before} be the number of secret key queries made before the challenge ciphertext.

Lemma 4.17. *Given Assumption 3',* **Hybrids** $k.1$ *and* $k.2$ *are indistinguishable for* $k \leq q_{before}$.

Proof. The only difference between **Hybrid** $k.1$ and **Hybrid** $k.2$ is in the $\mathfrak{R}_2$ component, and Assumption 3' gives us generators for all components. Therefore, we can focus on simulating $\mathfrak{R}_2$. Obtain the $\mathfrak{R}_2$ components of Assumption 3': $N = [1]_\ell, N' = [1]_{\ell+1}, \{P_i = [p_i]_{\ell+1}\}_{i \in [\ell+1]}, Q = [q]_{\ell(\ell+1)}$ where $q = \prod_{i \in [\ell+1]} p_i$ or q is random.

The public key has no $\mathfrak{R}_2$ components. The first $k-1$ secret keys have no $\mathfrak{R}_2$ components either, except $H^{\mathbf{u}}$ has a random $\mathfrak{R}_2$. Therefore, we can generate a random $\mu^{\mathbf{u}}$, and set $H^{\mathbf{u}} = \mu^{\mathbf{u}} N'^\ell$.

Now consider secret key k for identity $\mathbf{u}$. We will drop the $\mathbf{u}$ superscript in the secret key components for notational convenience. Remember that this key comes before the challenge ciphertext. We choose random $\alpha, \gamma', \delta, r', s_i$. Also choose random $\beta_{i,1-u_i}$, and set:

$$E_{i,1-u_i} = \beta_{i,1-u_i} s_i N' \qquad E_{i,u_i} = s_i P_i \qquad F_i = r' \beta_{i,1-u_i} s_i P_{\ell+1} \qquad G = \gamma' r' \prod_{i \in [\ell]} s_i N$$

$$H = \alpha N'^\ell + r' \prod_{i \in [\ell]} s_i \left(\delta P_{\ell+1} N'^{\ell-1} + Q \right)$$

In the case where $q = \prod_{i \in [\ell+1]} p_i$, this formally sets $\beta_{i,u_i} = p_i, r = r' p_{\ell+1}, \gamma = \gamma'/p_{\ell+1}$ in **Hybrid** $k.1$. In the case where q is random, this gives the same settings for the formal variables, but makes H random in $\mathfrak{R}_2$, as in **Hybrid** $k.2$. It remains to simulate the rest of the secret keys and the challenge ciphertext to be consistent with these formal variables. For secret keys after k, the $\mathfrak{R}_2$ component is empty except for $H^{\mathbf{u}}$, which contains α, so we can easily simulate these. For the challenge ciphertext, we will choose a random t', and our goal will be to set the formal variable t to be $t' p_{\ell+1}$. Thus, we need to compute

$$J^* = [\alpha\gamma' t']_{\ell(\ell+2)} \quad C^* = [\gamma' t']_{\ell} \quad D^* = \left[t' p_{\ell+1} \left(\delta + \sum_{\mathbf{v} \in S^*} \left(\prod_{i:v_i=u_i} p_i \right) \left(\prod_{i:v_i \neq u_i} \beta_{i,v_i} \right) \right) \right]_{\ell(\ell+1)}$$

Since we know $\alpha, \gamma', t', \delta$, we can generate most of the terms above ourselves. We only need to show how to generate the terms

$$C_{\mathbf{v}} = \left[p_{\ell+1} \left(\prod_{i:v_i=u_i} p_i \right) \left(\prod_{i:v_i \neq u_i} \beta_{i,v_i} \right) \right]_{\ell(\ell+1)}$$

for $\mathbf{v} \in S^*$. We will show how to compute these terms for any $\mathbf{v} \neq \mathbf{u}$, which is sufficient since $\mathbf{u} \notin S^*$. Since $\mathbf{v} \neq \mathbf{u}$, there is at least one j such that $v_j \neq u_j$. Therefore, we can compute

$$\left[p_{\ell+1} \prod_{i:v_i=u_i} p_i \right]_{(x+1)(\ell+1)}$$

from the P_i, where $x \leq \ell - 1$ is the number of bits $\mathbf{v}$ and $\mathbf{u}$ have in common. Then, if necessary, we can lift this to level $\ell(\ell+1)$ and multiply by $\prod_{i:v_i \neq u_i} \beta_{i,v_i}$ (which we know) to compute $C_{\mathbf{v}}$. This completes the simulation.

If an adversary can distinguish between **Hybrid** $k.1$ and **Hybrid** $k.2$, our reduction will therefore distinguish the two cases in Assumption 3', a contradiction. □

For $k > q_{before}$, the proof of Lemma 4.17 no longer applies, since we will not know the kth identity $\mathbf{u}$ until *after* the challenge query is made, meaning we do not know how to embed the Assumption 3' challenge into the challenge ciphertext. One possibility is to *guess* the identity $\mathbf{u}$, and abort if the guess was incorrect. While this works when the total number of users is polynomial, when we move to the identity-based setting, this results in an exponential loss in the security reduction. We therefore need an alternate approach to proving Lemma 4.14 for $k > q_{before}$.

Lemma 4.18. *Given Assumption 3',* **Hybrids** $k.1$ *and* $k.2$ *are indistinguishable for* $k > q_{before}$.

Proof. Like the proof of Lemma 4.17, we can focus on simulating $\mathfrak{R}_2$. Obtain the $\mathfrak{R}_2$ components of Assumption 3': $N = [1]_{\ell}, N' = [1]_{\ell+1}, \{P_i = [p_i]_{\ell+1}\}_{i \in [\ell+1]}, Q = [q]_{\ell(\ell+1)}$ where $q = \prod_{i \in [\ell+1]} p_i$ or q is random.

The public key has no $\mathfrak{R}_2$ components. The first q_{before} secret keys have no $\mathfrak{R}_2$ components either, except $H^{\mathbf{u}}$ has a random $\mathfrak{R}_2$. Therefore, we can generate a random $\mu^{\mathbf{u}}$, and set $H^{\mathbf{u}} = \mu^{\mathbf{u}} N'^{\ell}$.

Now consider generating the challenge ciphertext. We do not want to commit to the $\beta_{i,b}$ at this time, because we will then be unable to embed our challenge in the kth secret key query. Instead, we will generate the ciphertext in such a way that we will not commit to the values at this point. Choose random $\alpha, \gamma', \delta', t'$, and set the challenge ciphertext as

$$J^* = [\alpha\gamma' t']_{\ell(\ell+2)} \quad C^* = [\gamma' t']_{\ell} \quad D^* = [t'\delta']_{\ell(\ell+1)}$$

Our goal will be to formally set $\gamma = \gamma'/p_{\ell+1}, t = t' p_{\ell+1}$ and $\delta = \left(\delta' + \sum_{\mathbf{v} \in S^*} \prod_{i \in [\ell]} \beta_{i,b}\right) / p_{\ell+1}$, which will all be uniform random variables. The point is that, even though this is the formal setting

of variables we are targeting, we do not need to know the $\beta_{i,b}$ at this point. We simply need to make sure that when we generate components later, they are consistent with this setting of variables.

We generate secret key queries $q_{before}+1$ through $k-1$ as we did for queries before the challenge ciphertext. Secret key queries after the kth query have an empty $\mathfrak{R}_2$ component, except that $H^{\mathbf{u}}$ is an encoding of α, which we now know, so we can simulate these. It remains to simulate the kth secret key query on identity $\mathbf{u}$. We will drop the $\mathbf{u}$ superscript in the secret key components for notational convenience. Remember that this key comes after the challenge ciphertext, so we know S^*. We choose random r', s_i. Also choose random $\beta_{i,1-u_i}$. We wish to formally set $\beta_{i,u_i} = p_i$ and $r = r'p_{\ell+1}$ in **Hybrid** $k.1$. This amounts to generating the secret key as

$$E_{i,1-u_i} = [s_i\beta_{i,1-u_i}]_{\ell+1} \quad E_{i,u_i} = [s_i p_i]_{\ell+1} \quad F_i = [r'p_{\ell+1}s_i\beta_{i,1-u_i}]_{\ell+1} \quad G = [\gamma' r' \prod_{i\in[\ell]} s_i]_\ell$$

$$H = \left[\alpha + r' \prod_{i\in[\ell]} s_i \left(p_{\ell+1}\left(\delta' - \sum_{\mathbf{v}\in S^*}\left(\prod_{i:v_i=u_i} p_i\right)\left(\prod_{i:v_i\neq u_i}\beta_{i,v_i}\right)\right) + \prod_{i\in[\ell+1]} p_i\right)\right]_{\ell(\ell+1)}$$

We know $\alpha, \gamma', r', s_i, \delta'$, so we can easily simulate $E_{i,b}, F_i, G$. To simulate H, we generate the term involving α for ourselves, the $r'\prod_{i\in[\ell]} s_i p_{\ell+1}\delta'$ term from $P_{\ell+1}$, and the term involving $\prod_{i\in[\ell+1]} p_i$ from Q. Now we need to simulate the terms

$$C_{\mathbf{v}} = \left[p_{\ell+1}\left(\prod_{i:v_i=u_i} p_i\right)\left(\prod_{i:v_i\neq u_i}\beta_{i,v_i}\right)\right]_{\ell(\ell+1)}$$

We can compute $C_{\mathbf{v}}$ from the P_i exactly as in Lemma 4.17. If $q = \prod_{i\in[\ell+1]} p_i$, we simulate **Hybrid** $k.1$. If q is random, then the only difference is the $\mathfrak{R}_2$ component of $H^{\mathbf{u}}$ for the kth identity is random. Thus we simulate **Hybrid** $k.2$. Therefore, if an adversary can distinguish between **Hybrid** $k.1$ and **Hybrid** $k.2$, our reduction will distinguish the two cases in Assumption 3'. □

Acknowledgments

This work was supported by the DARPA PROCEED program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA.

References

[ABG+13] Prabhanjan Ananth, Dan Boneh, Sanjam Garg, Amit Sahai, and Mark Zhandry. Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689, 2013. http://eprint.iacr.org/2013/689.

[AGIS14] Prabhanjan Ananth, Divya Gupta, Yuval Ishai, and Amit Sahai. Optimizing obfuscation: Avoiding barrington's theorem. Cryptology ePrint Archive, Report 2014/222, 2014. http://eprint.iacr.org/2014/222.

[Att14] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 557–577, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany.

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

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275, Santa Barbara, CA, USA, August 14–18, 2005. 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.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300, Bengalore, India, December 1–5, 2013. Springer, Berlin, Germany.

[BWZ14] Dan Boneh, Brent Waters, and Mark Zhandry. Low overhead broadcast encryption from multilinear maps. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 206–223, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 480–499, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany.

[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.

[Del07] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215, Kuching, Malaysia, December 2–6, 2007. Springer, Berlin, Germany.

[DPP07] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *PAIRING 2007: 1st International Conference on Pairing-based Cryptography*, volume 4575 of *Lecture Notes in Computer Science*, pages 39–59, Tokyo, Japan, July 2–4, 2007. Springer, Berlin, Germany.

[FN93] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491, Santa Barbara, CA, USA, August 22–26, 1993. 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.

[GGH14] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graded multilinear maps from lattices. Cryptology ePrint Archive, Report 2014/645, 2014. http://eprint.iacr.org/2014/645.

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

[GGHZ14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure attribute based encryption from multilinear maps. Cryptology ePrint Archive, Report 2014/622, 2014. http://eprint.iacr.org/2014/622.

[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.

[GW09] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany.

[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.

[LSS14] Adeline Langlois, Damien Stehlé, and Ron Steinfeld. GGHLite: More efficient multilinear maps from ideal lattices. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 239–256, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany.

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

[Wee14] Hoeteck Wee. Dual system encryption via predicate encodings. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 616–637, San Diego, CA, USA, February 24–26, 2014. Springer, Berlin, Germany.

[Zha14] Mark Zhandry. How to avoid obfuscation using witness PRFs. Cryptology ePrint Archive, Report 2014/301, 2014. http://eprint.iacr.org/2014/301.