Adaptively Secure Broadcast Encryption with Small System Parameters

Mark Zhandry
Stanford University, USA
mzhandry@stanford.edu

Abstract

We build the first public-key broadcast encryption systems that simultaneously achieve adaptive security against arbitrary number of colluders, have small system parameters, and have security proofs that do not rely on knowledge assumptions or complexity leveraging. Our schemes are built from either composite order multilinear maps or obfuscation and enjoy a ciphertext overhead, private key size, and public key size that 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 non-falsifiable knowledge 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.

There are many ways to model the security of a broadcast scheme, the most desirable being called *adaptive* security. Here, the adversary may adaptively corrupt users, learning their secret keys, and then he chooses an arbitrary uncorrupted "challenge" set S^* of users that he wants to attack. At this point, the adversary is given a broadcast encrypted to S^*, and his goal is to learn some information about the plaintext broadcast. The adversary may continue corrupting users to achieve this goal, so long as he does not corrupt any of the intended recipients of the broadcast. Adaptive security best captures real-world settings; for example, the adversary may register many "dummy" users in the system, and then try to persuade or dissuade other users from subscribing for a broadcast based on the secret keys the adversary has acquired for the dummy users. We focus on adaptive security for this work.

Broadcast encryption admits a trivial adaptively-secure 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 secure based on the semantic security of the underlying public key encryption scheme.

Therefore, the main interest in broadcast encryption, besides security, is to minimize the parameter sizes. Minimizing ciphertext size is important for reducing the bandwidth required for broadcasting. Minimizing the size of the broadcast key and user secret keys is important for reducing the storage requirements for the broadcaster and receiver.

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, public keys have size proportional to the number of users N, secret keys are constant size, and the ciphertext overhead can be made proportional to the size $|S|$ of the recipient set. Using identity-based encryption (IBE), the public key size can even be reduced to constant size, and the resulting broadcast scheme is even identity-based[2].

In this work, we ask the following:

Can we construct adaptively-secure broadcast encryption with "short" ciphertexts, secret keys, and broadcast keys.

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 linear in N. One trade-off of their scheme is that it is only proved secure in a *static model*, where the adversary may still choose the challenge set S^* arbitrarily, but must choose it before corrupting any users and before even seeing the public parameters of the scheme. Some other schemes with constant-sized ciphertexts based on bilinear maps [DPP07, GW09] have been proven adaptively secure and/or are identity based, but 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. The first constructions of broadcast encryption from multilinear maps [BS02, GGH13a, CLT13, BW13] were all *secret key* schemes, where the broadcast key has to be kept secret. Moreover, for N users, these schemes require N-linear maps, resulting in systems super-linear secret keys[3].

Boneh, Waters, and Zhandry [BWZ14a] 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.

[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.

[2] The broadcast key would be the master public key for the system, and the secret for an identity is the secret key for the underlying IBE. To encrypt to a set S, simply encrypt separately to each identity in S.

[3] 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 $\omega(N)$

Existing 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$^+$13] show how to shrink the public key size in Boneh and Zhandry's scheme [BZ14], but at the cost of losing the distributed property, and security only being proved in the static model. Their scheme relies on stronger knowledge version of obfuscation, called differing inputs obfuscation (diO) [BGI$^+$01, BCP14, ABG$^+$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 also requires a strong knowledge version of witness PRFs, called extractable witness PRFs. Both diO and extractable witness PRFs are non-falsifiable assumptions. Moreover, Garg 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 (that is, the assumptions make explicit reference to the scheme).

Schemes with CCA security. Boneh, Gentry, and Waters [BGW05] show how to modify their scheme to be secure against a chosen ciphertext attack, where the adversary is additionally given access to a decryption oracle. The rough idea is to embed the verification key for a one-time signature into the ciphertext, and sign the resulting ciphertext using the corresponding signing key. Boneh, Waters, and Zhandry [BWZ14a] show that the same modification can be carried out on their low-overhead scheme. Phan et al. [PPSS12] give a different modification to [BGW05] that replaces the signature scheme with a universal one-way hash function, which can be more efficient than signatures. However, their proof implicitly requires that the representation of group elements is unique, and is therefore inapplicable to schemes based on current "noisy" multilinear maps[4].

Boneh et al. [BCHK07] show how to *generically* convert any identity-based (non-broadcast) encryption scheme into a chosen ciphertext attack (CCA)-secure public key encryption scheme using a similar high-level strategy. The idea is that the encrypter encrypts to the identity encoding the verification key (thus "embedding" the verification key into the ciphertext), and then signs the resulting ciphertext using the signing key as in [BGW05].

In the case of identity-based *broadcast* encryption, one may hope that the same conversion can be used. The idea would be that the encrypter adds to the recipient set the identity corresponding to a verification key $\mathsf{vk} \in \mathcal{VK}$ for a one-time signature, encrypts to the expanded set, and then signs the resulting ciphertext. Unfortunately, as observed in [PPSS12] this conversion is insufficient for proving chosen ciphertext security. For example, for the trivial scheme based on identity-based encryption, it is not difficult to show that the converted scheme is CCA-*insecure*. The reason is that, in the case of broadcast encryption, a decryption query allows the adversary to specify which user should decrypt the ciphertext[5]. The reduction *could* decrypt the ciphertext using the secret key corresponding to $\mathsf{vk}' \in \mathcal{VK}$, but the adversary may require decryption by a different user for which

[4]Indeed, their scheme is CCA-*in*secure using these multilinear maps

[5]For correctly generated ciphertexts, the correctness of broadcast encryption implies that all users decrypt the same way. However, improperly generated ciphertexts may decrypt differently for different users. Thus, decryption is not well-defined, unless a particular user is specified.

the reduction may not have a secret key. While for correctly generated ciphertexts, two users will always produce the same decryption, it is not necessarily true of malformed ciphertexts. Therefore, the reduction might not be able to determine the correct response to decryption queries.

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: the reduction guesses the adversary's challenge before seeing the public key, and will abort if the 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 a 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. Hence, the question of adaptive security for broadcast schemes is interesting even in the setting of sub-exponential hardness assumptions. Indeed, no existing work simultaneously achieved small parameters and adaptive security even relative to *sub-exponential* static hardness assumptions.

Developments in functional encryption. From a functionality perspective, broadcast encryption is a special case of *functional encryption*. In functional encryption, users can obtain secret keys corresponding to functions, and using a secret key for a function f on a ciphertext encrypting m yields $f(m)$. Intuitively, security says that given secret keys for functions $f_1, ..., f_n$, nothing can be learned about a plaintext except for $f_1(m), ..., f_n(m)$. Statically secure functional encryption was first built by Garg et al. [GGH+13b]. Subsequently, Waters [Wat14] and Ananth et al. [ABSV14] show how to obtain adaptively secure functional encryption from obfuscation, and Garg et al. [GGHZ14b] show how to obtain the same results from multilinear maps. However, the parameter sizes grow polynomially with the description size of f. This means that the resulting broadcast scheme, while adaptively secure, has parameters growing with polynomially with $|S|$. Thus, these results are not immediately useful in the setting of broadcast encryption.

1.1 Our Contributions

In this work, we give adaptively secure broadcast schemes where all parameters — ciphertexts, secret keys, and public keys — are poly-logarithmic in the number of users. As the number of users is polynomial, such poly-logarithmic terms can be bounded by the security parameter. Thus the parameter sizes can be taken to be independent of the number of users, which is the best possible. We give three main contributions:

- Our obfuscation-based scheme builds on the statically-secure schemes of Boneh and Zhandry [BZ14] and Ananth et al. [ABG+13]. The main idea behind those schemes is that the user has an obfuscated program that can decrypt the ciphertext only if the user provides a "proof" that she is allowed to decrypt. In the Boneh and Zhandry scheme, the program required as input a description of the recipient set S. This description is potentially as large as S, which results in a program size at least $|S|$. Ananth et al. show how to reduce the program size by hashing the description of S using a Merkle hash tree, but the resulting scheme can only be proved using the stronger notion of differing-inputs obfuscation (diO). Their scheme is still only statically secure.

  To restore security from iO while keeping programs short, we apply the techniques of Hubáček and Wichs [HW14]. Essentially, by replacing the Merkle hash with a *somewhere statistically binding* (SSB), security can be obtained using only iO. [HW14] constructs SSB hashes from fully homomorphic encryption. The resulting broadcast scheme is only statically secure, and we use additional techniques to obtain adaptive security.

  Along the way, we show a conversion from secret key functional encryption to public key functional encryption using obfuscation which preserves adaptive security. While Ananth et al. [ABSV14] show a similar result[6], ours is conceptually simpler and has more compact ciphertexts, which will be needed for the application to broadcast encryption.

- Our multilinear-map scheme is proved secure using polynomial reductions to simple falsifiable assumptions on composite-order symmetric multilinear maps. The scheme is based on the generically secure scheme of Boneh, Waters, and Zhandry [BWZ14a] (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 polylogarithmic 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 natural multilinear subgroup decision assumptions similar to those recently made by Garg et al. [GGHZ14a], as well as a multilinear DDH-style assumption. Due to lack of space, we present our multilinear map construction in Appendix 3.

- We also observe that if the output of decryption is statistically independent of the secret key used, then the CPA-to-CCA conversion sketched above goes through. While this conversion requires an identity-based scheme, we show a different conversion that leverages the broadcast property of the system (as apposed to the identity-based property), and therefore works for even non-identity-based broadcast schemes. We demonstrate that our adaptively secure schemes have the required statistically independent decryption, and we thus achieve full adaptive chosen ciphertext security under static assumptions.

We give a comparison of our work to existing works in Table 1.

Similar to [BWZ14a], we can set the number of users to be 2^λ for a security parameter λ, and obtain identity-based schemes that are both adaptively secure, and allows for an unbounded number of recipients with constant ciphertext size.

[6]Their conversion uses a staticaly-secure public key functional encryption scheme instead of obfuscation

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. All sizes hide multiplicative constants dependent on the security parameter (but not m or N). The column labeled "type" indicates which schemes are secret broadcast key (sk), public broadcast key (pk), or identity-based with public broadcast key (id). Finally, the "assump." column indicates which schemes are based on public key encryption (PKE), identity-based encryption (IBE), bilinear maps (BM), multilinear maps (MLM), or obfuscation-related primitives (Obf). "RO" denotes that the security proof is in the random oracle model, and "KNO" represents that the underlying assumptions are non-falsifiable knowledge assumptions

Scheme	$\|bk\|$	$\|ct\|$	$\|sk\|$	security	type	assump.
Trivial	N	$\|S\|$	1	adaptive	pk	PKE
	1	$\|S\|$	1		id	IBE
[BGW05]	N	1	1	static	pk	BM
	$\sqrt{N}$	$\sqrt{N}$	1			
[Del07]	m	1	1	static	id	BM
[GW09]	m	1	1	semi-static	pk	BM
	m	1	1	adaptive	id	BM (RO)
	$\sqrt{m}$	$\sqrt{\|S\|}$	1	adaptive		BM
([BS02] or [BW13]) and [GGH13a]	N^c	0	N^c	static	sk	MLM
[BWZ14a]	$\log^c N$	$\log^c N$	$\log^c N$	static	pk	MLM
	1	1	1		id	
[BZ14]	N^c	1	1	static	pk/id	Obf
[ABG+13]	$\log^c N$	1	1	static	pk	Obf (KNO)
[Zha14]	$\log^c N$	$\log^c N$	$\log^c N$	adaptive	pk	Obf (KNO)
This work	$\log^c N$	$\log^c N$	$\log^c N$	adaptive	pk	MLM or Obf
	1	1	1		id	

Instantiating our scheme. Our obfuscation-based scheme can be based on any one-way function, a somewhere statistically binding hash (which in turn can be build from FHE using [HW14]), and any indistinguishability obfuscator, such as the original candidate of Garg et al. [GGH$^+$13b].

For our multilinear map scheme, we present the construction and security proof assuming "ideal" multilinear maps in order to simplify the exposition and highlight our ideas. However, certain complications arise when using current multilinear map candidates such as [GGH13a]. Namely, current candidates are "noisy" and have a slightly different functionality than ideal maps. We explain how to adapt our construction an proof to these non-ideal maps.

A more serious issue with current multilinear candidates is the security of our computational assumptions: our assumptions belong to a class of "source group" assumptions that have proven problematic on current multilinear map candidates. In particular, due to a recent line of attacks [CHL$^+$14, GHMS14, BWZ14b, CLT14], our confidence of these assumptions on current maps is considerably diminished. In particular, the state-of-the-art — both in terms of constructions and attacks — is continuously changing, and the only viable candidate currently supporting these assumptions is the very recent candidate of Coron, Lepoint, and Tibouchi [CLT15]. Much more research is needed to gain confidence in these assumptions.

Since we describe our construction and assumptions generically, not relying on any specific candidate, our construction and proof can be ported to *any* multilinear map for which the assumptions hold (whether or not those maps are based on current candidates).

We also stress that, while our assumptions may be broken on many current candidates, the security of our scheme itself is still plausible. In particular, the assumption that the scheme is secure is actually a "target group" assumption on multilinear maps, for which the current attacks do not apply. Therefore, it is possible to instantiate our scheme on, say, a composite-order variant of the Garg-Gentry-Halevi [GGH13a] maps[7].

1.2 Technical Difficulties

Obfuscation-based scheme. As mentioned above, our obfuscation-based construction builds on the obfuscation-based protocols of Boneh and Zhandry [BZ14] and Ananth et al. [ABG$^+$13]. The rough idea in these schemes is that the ciphertext for a set S of users is obtained by XORing the message with the output of a pseudorandom function applied to S. To enable users in S to decrypt, the public parameters contain a public program that takes as input a set S, a user $u \in S$, and the secret key for user u, and outputs the pseudorandom function applied to S. There are several challenges in obtaining an adaptively secure scheme with small parameters from indistinguishability obfuscation (iO) using this strategy:

- During the security proof, Boneh and Zhandry hard-code S^* into the program, and the behavior of the program on this set is altered. Thus, S^* must be known at the setup time, which corresponds to the static security setting.

- Even specifying S^* requires space that grows linearly with the number of recipients, and so this altered program must be at least as large as S^*. To use iO (even iO for Turing machines), the altered program and the original program must be the same size. Thus, the public parameters must grow with the number of users. Ananth et al. get around this issue by instead having

[7][GGH13a] do not describe a composite-order version of their scheme, but it is straightforward to modify the scheme to support composite-order groups

the program take a hash h of S as input, and have the user supply a "proof" π that $u \in S$. Using Merkle hash trees, π can be made logarithmic in the number of users.

However, since the hash function must be compressing for h to be much smaller than S, false proofs exists, though they are hard to find. This necessitates using differing-inputs obfuscation (diO) to prove security. diO is a knowledge-type assumption, and in its simplest form seems implausible [GGHW14]. Avoiding the arguments in [GGHW14] usually requires making application-specific assumptions.

- A crucial step in Boneh and Zhandry's proof is to alter public parameters so that the secret key for users $u \in S^*$ do not exist. Again, this seems to require knowing S^* during setup time.

We circumvent the first issue above using a recent development by Hubáček and Wichs [HW14]. Namely, they describe a special type of hash function, called a *somewhere statistically binding* hash, which allows short proofs π as in Merkle hashing, but can also be used with iO.

To circumvent the second issue, we essentially place the obfuscated program *in the ciphertext*, instead of in the public parameters. This allows us to embed (a hash of) the set S^* into the program at the time of the challenge query, as needed to adaptive security.

However, we are still left with the third issue, namely that in order to use iO, we need to ensure that secret keys for users in S^* do not even exist (since such keys correspond to inputs that can decrypt the ciphertext. Even if such inputs are hard to find, using iO requires that they do not exist). The notion of "existence" of a secret key is relative to the public key, and therefore it seems that we need to know the set S^* when generating the public key at the beginning of the experiment, which again corresponds to static security. Note that this last issue is only present in the public key setting: in the secret key setting, no key is provided to the adversary at the beginning of the experiment. We therefore construct a *single-ciphertext* secret key broadcast system from obfuscation, where the "existence" property of a secret key is defined relative to the (single) ciphertext. When the challenge ciphertext is generated, the set S^* is known, so we can construct the ciphertext to guarantee that no secret keys for users in S^* exist.

Finally, we present a transformation from such a single-ciphertext secret key broadcast scheme into a public key (many-time) scheme using obfuscation. This transformation preserves adaptivity and roughly preserves parameter sizes, so that the obtained scheme obtains adaptive security with short system parameters. As mentioned above, this conversion can also be applied to functional encryption, obtaining adaptively secure public-key functional encryption from adaptively secure single-ciphertext secret key functional encryption. Thus, we offer an alternative to the work Ananth et al. [ABSV14]. Our conversion uses obfuscation and is conceptually very simpler, whereas theirs uses the weaker primitive of statically secure public key functional encryption but is conceptually a bit more complicated.

In the case of functional encryption, ciphertext size is not a concern, and single-ciphertext secret key functional encryption can be built from one-way functions. Piecing together, we obtain another adaptively-secure public key functional encryption scheme from obfuscation and one-way functions, offering a very simple alternative to the schemes of Waters [Wat14] and Ananth et al. [ABSV14].

Multilinear map-based scheme. Our multilinear map scheme is proven secure using multilinear analogs of the dual system framework [Wat09, Wee14]. 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 (but still polynomial) 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 $S' \neq S$ such that ct is also 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 for this "information-theoretic" step 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 *security 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, our scheme has elements in different levels that are highly correlated, and we would need to simulate these correlations using the challenge elements. 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, which we wish to avoid.

Instead, we continue to use symmetric multilinear maps, but modify the scheme by introducing more correlations so that the new 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 style of assumption we use.

This alone is not enough, however, because in the other computational steps, we can only produce the needed correlations by carrying out the multilinear pairing operation. Increasing the multilinearity naively to allow the reduction to simulate these correlations results in fewer of the DDH-style assumptions holding (for example, $(g, g^a, g^b, g^c, g^{abc})$ *can* be distinguished from (g, g^a, g^b, g^c, g^d) on a tri-linear map). Therefore, we need to introduce even more correlations into our scheme so that the required DDH-style assumption holds, setting off a vicious cycle. Put another way, 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 [BWZ14a]. However, the number of pairing operations required by our scheme is comparable to the BWZ scheme. In current multilinear maps, it is the number of multiplications that determines the parameter sizes, so the *effective* multilinearity required by our scheme is essentially the same as in the BGW scheme.

The second 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. [GGHZ14a] 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 $(\mathsf{params}, \mathsf{msk})$ outputted by $\mathsf{Setup}(\mathcal{ID})$, every set $S \subseteq \mathcal{ID}$, every sk_u outputted 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 $u \in [N]$, 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 $\mathcal{A}$.

CCA Queries: $\mathcal{A}$ may adaptive make chosen ciphertext queries on header Hdr, set S and user $u \in S$. The challenger responds by computing $\mathsf{sk}_u \leftarrow \mathsf{KeyGen}(\mathsf{msk}, u)$, and then returning $K \leftarrow \mathsf{Dec}(\mathsf{params}, u, \mathsf{sk}_u, S, \mathsf{Hdr})$

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 $u \notin S^*$

More CCA queries: $\mathcal{A}$ may continue making CCA queries, conditioned on $(\mathsf{Hdr}, S) \neq (\mathsf{Hdr}^*, S^*)$.

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 $\texttt{EXP}(b)$. We define the adaptive advantage of $\mathcal{A}$, as $\texttt{BE}_{\mathcal{A}}^{(\text{adv})} = |\Pr[\texttt{W}_0] - \Pr[\texttt{W}_1]|$

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

We can also define several weaker variants. CPA security is obtained if we prevent the adversary from making CCA queries. Static (CCA or CPA) security is obtained by requiring the adversary to commit to S^* before seeing the public parameters params. The intermediate notion of semi-static (CCA or CPA) security is obtained by requiring the adversary to commit to a set $\hat{S}$ before seeing the public parameters, and then restricting secret key queries to be on users $u \notin \hat{S}$ and the challenge query to be on a set $S^* \subseteq S$. For this work, we will be mainly focusing on adaptive CPA and CCA security.

Statistically independent decryption. Beyond correctness and security, another property we are interested in this work is called *statistically independent decryption*. This means that the output of decryption does not depend on the user who decrypts. Of course, for correctly generated ciphertexts, the choice of user is irrelevant due to the correctness of the scheme. However, we wish this property to hold for even improperly generated ciphetexts. More precisely, we allow for a randomized decryption procedure, and require, with overwhelming probability over the choice of $(\mathsf{params}, \mathsf{msk}) \leftarrow \mathsf{Setup}()$, that for all users u_0, u_1, all sets S with $u_0, u_1 \in S$ and all headers Hdr (even invalid ones) the following distributions on K are statistically close:

$$K \leftarrow \mathsf{Dec}(\mathsf{params}, u_0, \mathsf{sk}_{u_0}, S, \mathsf{Hdr}) \text{ where } \mathsf{sk}_{u_0} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, u_0) \text{ and}$$
$$K \leftarrow \mathsf{Dec}(\mathsf{params}, u_1, \mathsf{sk}_{u_1}, S, \mathsf{Hdr}) \text{ where } \mathsf{sk}_{u_1} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, u_1)$$

3 Multilinear Map-Based Construction

3.1 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 [GGHZ14a], 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[8] 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 $\alpha \times [\beta]_i = [\alpha]_0 \times [\beta]_i = [\alpha\beta]_i$. Therefore, even though it is not possible to publicly compute $[\alpha]_i$ for a given $\alpha \in \mathfrak{R}$, once given an encoding $[\beta]_i$, it is possible to compute the encoding $[\alpha\beta]_i$.

$\mathsf{ext}(\mathsf{Params}, e)$: The public extraction procedure takes a level t encoding e, and extracts a λ-bit string s from e. We require that, for any $i \in [k]$, if α is sampled such that the $\mathfrak{R}_i$ component is uniform and independent of the other components, 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:

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

To instantiate multilinear maps, we can use the Boneh, Wu, and Zimmermans modification [BWZ14b] to the Coron-Lepoint-Tibouchi (CLT) multilinear maps [CLT13]. This modification is designed to emulate multilinear groups of composite order, and to allow assumptions regarding subgroups of the multilinear groups[9]. However, current candidate multilinear maps, including the Boneh-Wu-Zimmerman 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

[8]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.

[9]The Garg-Gentry-Halevi [GGH13a] and basic Coron-Lepoint-Tibouchi [CLT13] multilinear maps do not satisfy these subgroup assumptions.

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.

- 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 be multiplied by higher-level encodings: for example, $[\alpha]_0$ can be multiplied by $[\beta]_i$ to obtain an encoding $[\alpha\beta]_i$. For our scheme, this functionality will be sufficient.

- Finally, in CLT encodings, and thus Boneh-Wu-Zimmerman 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 a particular 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.

3.2 The 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 [BWZ14a] (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[10]. Draw a random $\alpha, \gamma, \delta, \sigma \in \mathfrak{R}$ and for $i \in [\ell]$ and $b \in \{0,1\}$, draw random $\beta_{i,b} \in \mathfrak{R}$. The public key is

$$\mathsf{pk} = \left(\mathsf{Params}, \left\{A_{i,b} = [\beta_{i,b}]^1_{\ell+1}\right\}_{i\in[\ell], b\in\{0,1\}}, B = [\gamma]^1_\ell, L = [\delta]^1_{\ell(\ell+1)}, I = [\alpha\gamma]^1_{\ell(\ell+2)}, M = [\sigma]^3_\ell\right)$$

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

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

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 $s^\mathbf{u}_i \in \mathfrak{R}$ for $i \in [\ell]$. Also pick random $\eta_{i,b}, \zeta_i, \xi, \mu \in \mathfrak{R}$. Output the secret key components

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

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

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

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

for any $\mathbf{v} \neq \mathbf{u}$ where $\phi_{\mathbf{v}}^{\mathbf{u}}$ is some ring element as follows. Choose some i^* such that $v_{i^*} \neq u_{i^*}$. 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 $r^{\mathbf{u}} s_{i^*}^{\mathbf{u}} \beta_{i^*,1-u_{i^*}} = r^{\mathbf{u}} s_{i^*}^{\mathbf{u}} \beta_{i^*,v_{i^*}}$. Therefore, compute

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

$\mathsf{Dec}(\mathsf{sk}^{\mathbf{u}}, S, (C, D))$: Our decryption procedure is randomized. Choose a random $\rho, \xi', \mu' \in \mathfrak{R}$, and compute

$$G' = G^{\mathbf{u}} + \rho B + \xi' M = \left[\gamma(\rho + r^{\mathbf{u}} \prod_{i\in[\ell]} s_i^{\mathbf{u}}) \right]_{\ell}^1 + [\xi'\sigma + \xi]_{\ell}^3$$

$$\begin{aligned} H' &= H^{\mathbf{u}} + \sum_{\mathbf{v}\in S\setminus\{\mathbf{u}\}} Z_{\mathbf{v}}^{\mathbf{u}} + \rho \left(L + \sum_{\mathbf{v}\in S} \prod_{i\in[\ell]} A_{i,v_i} \right) + \mu' M^{\ell+1} \\ &= [\alpha]_{\ell(\ell+1)}^{1,2} + \left[\left(\rho + r^{\mathbf{u}} \prod_{i\in[\ell]} s_i^{\mathbf{u}} \right) \left(\delta + \sum_{\mathbf{v}\in S} \prod_{i\in[\ell]} \beta_{i,v_i} \right) \right]_{\ell(\ell+1)}^1 + \left[\mu'\sigma^\ell + \mu \right]_{\ell(\ell+1)}^3 \end{aligned}$$

Then compute

$$J' = H' \cdot C - D \cdot G'$$

Correctness. It suffices to show that (1) $J' = J$ in decryption and (2) G', H' are statistically independent of the user $\mathbf{u}$ performing decryption. The statistical independence of decryption on $\mathbf{u}$ follows from G', H' being statistically independent of $\mathbf{u}$.

First, define $\rho' = \rho + r^{\mathbf{u}} \prod_{i\in[\ell]} s_i^{\mathbf{u}}$. Since ρ is uniformly random in $\mathfrak{R}_1$, so is ρ'. Moreover, $\xi'\sigma + \xi$ and $\mu'\sigma^\ell + \mu$ are uniformly random in $\mathfrak{R}_3$. Therefore, the $\mathfrak{R}_1$ and $\mathfrak{R}_3$ distributions of G', H' are independent of $\mathbf{u}$. Moreover, the $\mathfrak{R}_2$ components are fixed (as 0 and α, respectively). Therefore, the distributions of G', H' are independent of $\mathbf{u}$.

Notice that since C and D have no $\mathfrak{R}_2$ or $\mathfrak{R}_3$ component, the result J' will only have a $\mathfrak{R}_1$ component. Therefore, we have

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

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. In order to ensure that G', H' are statistically independent of $\mathbf{u}$, we will need to re-randomized G', H' before computing J'. 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 $\ell+1$ for computing secret key encodings. 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 $\mathsf{poly}(\log |S|, \ell) = \mathsf{polylog}(|\mathcal{ID}|)$.

Next, we will prove the security of our scheme.

3.3 Security Proof for Multilinear Map-Based Construction

3.4 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. [GGHZ14a], 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 3.2 (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 3.3 (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 3.4 (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 three alternate 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 3.5 (Assumption 1')**.** For any ℓ, t, for a t-linear map, no efficient adversary can distinguish the following two distributions, where m, n, o, 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 3.6. *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 3.7 (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 3.8. *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. $\square$

Definition 3.9 (Assumption 3'). For any ℓ, t, for a t-linear map, no efficient adversary can distinguish the following two distributions, where $j = \lceil (t+1)/(\ell+1) \rceil$:

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

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

Lemma 3.10. *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 $P_i = [p_i]_1^2$ and $Q = [\prod_{i\in[t+1]} p_i]_1^2$ 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 $P_i = p_i N = [p_i]_1^2$. 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 $Q = [\prod_{i\in[t+1]} p_i]_1^2$, 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. $\square$

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.

3.5 Security Theorem and Proof

Here we prove the adaptive CPA security of our scheme.

Theorem 3.11. *If Assumptions 1', 2', and 3' hold, then our scheme is adaptively CPA 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 challenge ciphertext is now generated in the subring $\mathfrak{R}_1 \times \mathfrak{R}_2$, as opposed to $\mathfrak{R}_1$ as in **Hybrid** *Real*. The challenge ciphertext is:

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

and $K^* = \mathsf{ext}(\mathsf{Params}, J^*)$. The following is proved in Section 3.5.1:

Lemma 3.12. *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. In a semi-funcitonal secret key, the $\mathfrak{R}_2$ component of $H^{\mathbf{u}}$ is a random ring element, as opposed to encoding α as in a functional secret key. More precisely, a semi-functional key is generated as $\mathsf{sk}^{\mathbf{u}} = (\left\{E^{\mathbf{u}}_{i,b}\right\}_{i\in[\ell],b\in\{0,1\}}, \{F^{\mathbf{u}}_i\}_{i\in[\ell]}, G^{\mathbf{u}}, H^{\mathbf{u}})$ 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 3.13. *Given Assumptions 2' and 3',* **Hybrid** $k-1$ *and* **Hybrid** k *are indistinguishable.*

Proving 3.13 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.* In a correlated semi-functional secret key, all secret key components are generated in $\mathfrak{R}_1 \times \mathfrak{R}_2$. 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 3.5.2:

Lemma 3.14. *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 3.15. *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 3.5.3, we show how to handle both cases, each with a single invocation of Assumption 3'.

To finish the proof of Lemma 3.13, we need to prove the following, proved in Section 3.5.2:

Lemma 3.16. *Given Assumption 2',* **Hybrid** $\ell.2$ *and* **Hybrid** ℓ *are indistinguishable.*

Notice that the only difference between **Hybrid** $\ell.2$ and **Hybrid** ℓ is that the ℓth secret key is not semi-functional. This means that the $\mathfrak{R}_2$ components of all terms except $H^{\mathbf{u}}$ are empty.

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 3.17. *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 3.11. Now we fill in the proofs of the hybrids.

3.5.1 Proof of Lemma 3.12

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[11].

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

$$E_{i,b}^{\mathbf{u}} = s_i^{\mathbf{u}} \cdot A_{i,b} + \eta'_{i,b} O' \qquad F_i^{\mathbf{u}} = r^{\mathbf{u}} s_i^{\mathbf{u}} A_{i,1-u_i} + \zeta'_i O'$$

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

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:

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

[11]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.

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. □

3.5.2 Proof of Lemma 3.14 and Lemma 3.16

Proof of Lemma 3.14. 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 3.12, 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: $j < k$, $j = k$, and $j > k$. Secret keys for $j > k$ are also simulated exactly as in the proof of Lemma 3.12. Secret keys for $j < k$ are simulated as:

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

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

It is straightforward to see that this is the correct simulation. Secret key $j = k$ 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 3.16. The proof is almost identical to the proof of Lemma 3.14, 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. □

3.5.3 Proof of Lemma 3.15

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

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

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

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 3.18 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 3.15 for $k > q_{before}$.

Lemma 3.19. *Given Assumption 3',* **Hybrids** *$k.1$ and $k.2$ are indistinguishable for $k > q_{before}$.*

Proof. Like the proof of Lemma 3.18, 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

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

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

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

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

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 3.18. 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'. $\square$

4 Obfuscation-Based Construction

4.1 Primitives Needed

Indistinguishability Obfuscation. An *indistinguiability obfuscator* iO for a circuit class $\{\mathcal{C}_\lambda\}$ is a PPT uniform algorithm satisfying the following conditions:

- $\mathsf{iO}(\lambda, C)$ preserves the functionality of C. That is, for any $C \in \mathcal{C}_\lambda$, if we compute $C' = \mathsf{iO}(\lambda, C)$, then $C'(x) = C(x)$ for all inputs x.

- For any λ and any two circuits $C_0, C_1 \in \mathcal{C}_\lambda$ with the same functionality, the circuits $\mathsf{iO}(\lambda, C)$ and $\mathsf{iO}(\lambda, C')$ are indistinguishable. More precisely, for all pairs of PPT adversaries $(Samp, D)$ there exists a negligible function α such that, if $\Pr[\forall x, C_0(x) = C_1(x) : (C_0, C_1, \sigma) \leftarrow Samp(\lambda)] > 1 - \alpha(\lambda)$, then $\left|\Pr[D(\sigma, \mathsf{iO}(\lambda, C_0)) = 1] - \Pr[D(\sigma, \mathsf{iO}(\lambda, C_1)) = 1]\right| < \alpha(\lambda)$

The circuit classes we are interested in are polynomial-size circuits — that is, when $\mathcal{C}_\lambda$ is the collection of all circuits of size at most λ. We call an obfuscator for this class an *indistinguishability obfuscator for P/poly*. The first candidate construction of such obfuscators is due to Garg et al. [GGH+13b].

When clear from context, we will often drop λ as an input to iO.

Puncturable PRFs. A puncturable PRF is a PRF that can be "punctured" at an input, allowing the computation of the PRF at all points except the punctured input. More formally, for sets $\mathcal{X}, \mathcal{Y}$, a punctured PRF is a pair of algorithms $(\mathsf{Gen}, \mathsf{Punc})$ where:

- Gen outputs an efficiently computable function $\mathsf{F} : \mathcal{X} \to \mathcal{Y}$

- $\mathsf{Punc}(\mathsf{F}, x)$ takes as input a function F outputted by Gen and an input $x \in \mathcal{X}$, and outputs a "punctured" function F^x such that

$$\mathsf{F}^x(x') = \begin{cases} \mathsf{F}(x') & \text{if } x' \neq x \\ \perp & \text{if } x' = x \end{cases}$$

For security, we require that, for any input $x \in \mathcal{X}$, $y = \mathsf{F}(x)$ is indistinguishable from a random element in $\mathcal{Y}$, even given the punctured function $\mathsf{F}^x = \mathsf{Punc}(\mathsf{F}, x)$. The original GGM construction [GGM86] is an example of a puncturable PRF.

Somewhere statistically binding hash. We will also be relying on the work of Hubáček and Wichs [HW14], namely their new tool called somewhere statistically binding hash functions (SSB Hash). SSH Hashes where used in [HW14] as a special type of collision resistant hash function that can be used with indistinguishability obfuscation. Let $\Sigma, \mathcal{Z}$ be sets. Recall from [HW14] that a SSB Hash is a triple of algorithms $(\mathsf{Gen}, \mathsf{Open}, \mathsf{Ver})$ where:

- $\mathsf{Gen}(s, i)$ takes as input two integers s and i, where s denotes the number of blocks that will be hashed, and $i \in [s]$ indexes a particular block. The output is a function $H : \Sigma^s \to \mathcal{Z}$. The size of the descrption of H is independent of s and i (though it will depend on the security parameter).
- $\mathsf{Open}(H, x = \{x_\ell\}_{\ell \in [s]}, j)$ for $x_\ell \in \Sigma$ and $j \in [s]$ produces an "opening" π that "proves" that the jth element in x is x_j.
- $\mathsf{Ver}(H, h \in \mathcal{Z}, j \in [s], u \in \Sigma, \pi)$ either accepts or rejects. The idea is that Ver should only accept when $h = H(x)$ where $x_j = u$.
- Correctness: $\mathsf{Ver}(H, H(x), j, x_j, \mathsf{Open}(H, x, j))$ accepts.
- Index hiding: $\mathsf{Gen}(s, i_0)$ is computationally indistinguishable from $\mathsf{Gen}(s, i_1)$ for any i_0, i_1.
- Somewhere Statistically Binding: If $H \leftarrow \mathsf{Gen}(s, i)$, then if $\mathsf{Ver}(H, h, i, u, \pi)$ and $\mathsf{Ver}(H, h, i, u', \pi')$ accept, it must be that $u = u'$.

4.2 Secret Key to Public Key Conversion

First, we show a conversion from secret key broadcast encryption to public key broadcast encryption using obfuscation.

Let $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ be a *secret key scheme* (that is, the master secret key msk' is required to encrypt, and there is no public parameters). The security notion that is of interest to us is called 1-ciphertext security, which corresponds to the security notion in Section 2 with only a single challenge. While in the public key setting, such a notion implies security when there are multiple challenges, this is not the case in the symmetric setting since users cannot encrypt messages for themselves. Nonetheless, the single challenge security is sufficient for our purposes.

Construction 4.1. Let G be a PRG that maps $\mathcal{X}$ into $\mathcal{Y}$ with $|\mathcal{Y}| \gg |\mathcal{X}|$. Let F be a puncturable PRF from $\mathcal{Y}$ into $\mathcal{R}$, where $\mathcal{R}$ is the space of random coins used by Setup'.

$\mathsf{Setup}(\mathcal{ID})$ Choose a random F as the master secret key. The public key is $\mathsf{pk} = \widehat{P_{\mathsf{Enc}}} = \mathsf{iO}(P_{\mathsf{Enc}}^{\mathsf{F}})$ where $P_{\mathsf{Enc}}^{\mathsf{F}}$ is the program given in Figure 1.

$\mathsf{Enc}(\mathsf{pk}, S)$ Choose a random $s \leftarrow \mathcal{X}$ and let $r = \mathsf{G}(s)$. Run $\mathsf{msk}'_r = \widehat{P_{\mathsf{Enc}}}(s)$. Then compute $(K, \mathsf{Hdr}') = \mathsf{Enc}'(\mathsf{msk}'_r, S)$. The header is $\mathsf{Hdr} = (r, \mathsf{Hdr}')$ and the message encryption key is K.

$\mathsf{KeyGen}(\mathsf{msk}, u)$ Compute and output $\mathsf{sk}_u = \widehat{P_{\mathsf{sk}}} = \mathsf{iO}(P_{\mathsf{sk}}^{\mathsf{F}_{main}, u})$ where $P_{\mathsf{sk}}^{\mathsf{F}_{main}, u}$ is the program in Figure 2.

$\mathsf{Dec}(\mathsf{sk}_u, S, \mathsf{Hdr})$ Run $\mathsf{sk}'_u = \widehat{P_{\mathsf{sk}}}(r)$ and then $\mathsf{Dec}'(\mathsf{sk}'_u, S, \mathsf{Hdr}')$

Correctness. Correctness follows easily from the correctness of the underlying symmetric scheme.

Inputs: s
Constants: F

1. Let $r = \mathsf{G}(s)$
2. Output $\mathsf{msk}'_r = \mathsf{Setup}'(\mathsf{F}(r))$

Figure 1: The program $P^{\mathsf{F}}_{\mathsf{Enc}}$.

Inputs: r
Constants: F, u

1. $\mathsf{msk}'_r = \mathsf{Setup}'(\mathsf{F}(r))$
2. Output $\mathsf{KeyGen}'(\mathsf{msk}'_r, u)$

Figure 2: The program $P^{\mathsf{F},u}_{sk}$.

Parameter sizes. Let $|\mathsf{msk}'|, |\mathsf{sk}'|, |\mathsf{Hdr}'|$ be the sizes of the master secret key, user secret keys, and headers, respectively, in the secret key scheme, and $|\mathsf{msk}|, |\mathsf{pk}|, |\mathsf{sk}|, |\mathsf{Hdr}|$ be the parameter sizes of the resulting public key scheme. Let $t(\mathsf{KeyGen}')$ and $t(\mathsf{Setup}')$ be the running times of KeyGen' and Setup', respectively. Then, ignoring the security parameter,

$$|\mathsf{msk}| = O(1) \qquad |\mathsf{pk}| = t(\mathsf{Setup}')^{O(1)}$$
$$|\mathsf{sk}| = \left(t(\mathsf{KeyGen}') + t(\mathsf{Setup}')\right)^{O(1)} \qquad |\mathsf{Hdr}| = |\mathsf{Hdr}| + O(1)$$

Thus if $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ is low-overhead and has efficient setup and key generation, then so does $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$.

Application to functional encryption. The above conversion can be trivially extended to boost adaptively secure 1-ciphertext many-key secret key functional encryption into adaptively secure public key functional encryption. As observed by Ananth et al. [ABSV14], such secret key schemes can be built from one-way functions using known techniques. Thus, we obtain another construction of adaptively secure functional encryption from obfuscation. We believe ours is conceptually simpler than the schemes of Waters [Wat14] and Ananth et al. [ABSV14], though the conversion of [ABSV14] has the advantage of requiring weaker primitives than obfuscation.

Theorem 4.2. *If* $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ *is 1-ciphertext adaptively CCA (resp. CPA) secure,* F *is a secure puncturable PRF,* G *is a secure PRG, and* iO *is a secure indisitnguishability obfuscation, then* $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ *in Construction 4.1 is adaptively CCA (resp. CPA) secure.*

Proof. We prove security through a sequence of hybrid games.

Hybrid 0**.** This is the normal game where, on challenge set S^*, the adversary receives the header $\mathsf{Hdr}^* = (r^*, \mathsf{Hdr}'^*)$, and either the correct message encryption key K^*, or a random key. Let ϵ be advantage the adversary has in guessing which key he is given. We can assume $r^* = \mathsf{G}(s^*)$ is sampled at the beginning of the experiment, and $\mathsf{msk}'^* = \mathsf{Setup}'(\mathsf{F}(r^*))$ is computed directly from F rather than through $\widehat{P_{\mathsf{Enc}}}$.

Hybrid 1. This is the same as **Hybrid** 0, except that r^* is replaced with a uniformly random string. The security of G shows that this modification is undetectable.

Hybrid 2. This is the same as **Hybrid** 1, except that F is punctured at r^*, obtaining F^{r^*}, and $\widehat{P_{\mathsf{Enc}}}$ is replaced with an obfuscation of $P_{\mathsf{Enc}}^{r^*,\mathsf{F}^{r^*}}$ given in Figure 3. Since G is expanding, with overwhelming probability r^* is not in the image of G, and therefore the programs $P_{\mathsf{Enc}}^{r^*,\mathsf{F}^{r^*}}$ and $P_{\mathsf{Enc}}^{\mathsf{F}}$ are functionally equivalent. Thus, by the security of iO, this change is undetectable.

```
Inputs: $s$
Constants: $r^*$,$\mathsf{F}^{r^*}$

  1. Let $r = \mathsf{G}(s)$
  2. If $r = r^*$, abort and output $\perp$.
  3. Output $\mathsf{msk}'_r = \mathsf{Setup}'(\mathsf{F}(r))$
```

Figure 3: The program $P_{\mathsf{Enc}}^{r^*,\mathsf{F}^{r^*}}$.

Hybrid 3. This is the same as **Hybrid** 2, except that, for each secret key query on user u, first $\mathsf{sk}_u^* = \mathsf{KeyGen}'(\mathsf{msk}'^*, u)$ is computed, and the $\widehat{P_{\mathsf{sk}}}$ is replaced with an obfuscation of $P_{\mathsf{sk}}^{r^*,\mathsf{F}^{r^*},u,\mathsf{sk}_u^*}$ given in Figure 4. Notice that the programs $P_{\mathsf{sk}}^{r^*,\mathsf{F}^{r^*},u,\mathsf{sk}_u^*}$ and $P_{\mathsf{sk}}^{\mathsf{F},u}$ are functionally equivalent for this coice of sk_u^*, so the security of iO shows that these changes are undetectable.

Hybrid 4. This is the same as **Hybrid** 3, except that msk'^* is chosen freshly at random from Setup', instead of as $\mathsf{Setup}'(\mathsf{F}(r^*))$. This amounts to replacing $\mathsf{F}(r^*)$ with a fresh random string. Since the entire security game, except for the generation of msk'^*, only uses the punctured PRF F^{r^*}, PRF security shows that this change is undetectable.

Now we observe that **Hybrid 4** can be simulated using the challenger for $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$. Namely, at the begining of the experiment, we give the adversary an indistinguishability obfuscation of $P_{\mathsf{Enc}}^{r^*,\mathsf{F}^{r^*}}$ for a random r^* and random PRF F. On the adversary's challenge query, we forward the

```
Inputs: $r$
Constants: $r^*$,$\mathsf{F}^{r^*}$, $u$, $\mathsf{sk}_u^*$

  1. If $r = r^*$, then abort and output $\mathsf{sk}_u^*$.
  2. $\mathsf{msk}'_r = \mathsf{Setup}'(\mathsf{F}(r))$
  3. Output $\mathsf{KeyGen}'(\mathsf{msk}'_r, u)$
```

Figure 4: The program $P_{sk}^{r^*,\mathsf{F}^{r^*},u,\mathsf{sk}_u^*}$.

query to the challenger, obtaining Hdr'^*, K^* where K^* is the correct key for the secret key scheme or a random key. Then we give the adversary $\mathsf{Hdr}^* = (r^*, \mathsf{Hdr}'^*)$. When the adversary makes a secret key query for a user u, we make the same query to the challenger, obtaining sk_u^*, which we then use to compute the obfuscation of $P_{\mathsf{sk}}^{r^*, \mathsf{F}^{r^*}, u, \mathsf{sk}_u^*}$. When the adversary makes a guess, we output that guess.

Our advantage is identical to the advantage of the adversary in **Hybrid 4**, and must be negligible by the security of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$. Therefore, the adversary's advantage in **Hybrid 0** must also have been negligible.

□

4.3 A Secret Key Scheme

We now present our 1-ciphertext secret key scheme, which combined with the conversion above gives a public key scheme with small parameters.

Construction 4.3. Let F be a puncturable PRF from $\mathcal{ID}$ into $\mathcal{X}$. Let $(\mathsf{Gen}, \mathsf{Open}, \mathsf{Ver})$ be a SSB hash function with $\Sigma = \mathcal{ID}$.

$\mathsf{Setup}(\mathcal{ID})$ Choose a random F as the master secret key.

$\mathsf{Enc}(\mathsf{msk}, S)$ Choose a random message encryption key K. Run $H \leftarrow \mathsf{Gen}(|S|, 1)$ and compute $H(S)$ where S is treated as a list. Let $\widehat{P_{ciph}} = \mathsf{iO}(P_{ciph}^{\mathsf{F},H,h,K})$ where $P_{ciph}^{\mathsf{F},H,h,K}$ is the program in Figure 5. The header is $\mathsf{Hdr} = \widehat{P_{ciph}}$.

$\mathsf{KeyGen}(\mathsf{msk}, u)$ Output $\mathsf{F}(u)$

$\mathsf{Dec}(\mathsf{sk}_u, S, \mathsf{Hdr})$ Let ℓ be the index of u in S when S is sorted (if $u \notin S$, abort and output $\perp$). Compute $(h, \pi) \leftarrow \mathsf{Open}(H, S, \ell)$. Now run $K' \leftarrow \widehat{P_{ciph}}(\ell, u, \pi, \mathsf{sk}_u)$. Output K'.

Inputs: ℓ, u, π, t
Constants: F, H, h, K

1. Run $\mathsf{Ver}(H, h, \ell, u, \pi)$ to check that π is an opening of h at index ℓ to identity u. If the check fails, abort and output $\perp$.
2. Check that $\mathsf{G}(t) = \mathsf{G}(\mathsf{F}(u))$. If the check fails, abort and output $\perp$.
3. Output K.

Figure 5: The program $P_{ciph}^{\mathsf{F},H,h,K}$.

In Section 4.4, we explain how to modify the scheme to make decryption statistically independent of the user secret key.

Theorem 4.4. *If* F *is a secure puncturable PRFs,* $(\mathsf{Gen}, \mathsf{Open}, \mathsf{Ver})$ *is a secure somewhere statistically binding hash, and* iO *is a secure indistinguishability obfuscator, then Construction 4.3 is an adaptively secure 1-ciphertext secret key broadcast encryption scheme.*

Proof. We prove security through a sequence of hybrid games.

Hybrid 0. This is the normal game where, on challenge set S^*, the adversary receives the header $\mathsf{Hdr}^* = (H^*, \widehat{P_{ciph}}^*)$, and either the correct message encryption key K^*, or a random key. Let ϵ be advantage the adversary has in guessing which key he is given.

Let q_{before} be the number of secret key queries made before the challenge ciphertext. WE define the following hybrids for $i = 0, ..., q_{before}$:

Hybrid i. This is the same as **Hybrid 0**, except that on the first i secret key queries, the secret key given is just a uniform random string.

We now wish to show that **Hybrid** $(i-1)$ is indistinguishable from **Hybrid** i for each $i = 1, ..., q_{before}$. We do this through a sequence of hybrids.

Hybrid $i.1$ This is the same as **Hybrid** $(i-1)$, except that $\widehat{P_{ciph}}^*$ is replaced with $\mathsf{iO}(P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*})$ where F^{u_i} is F punctured at the point u_i (the ith secret key query), and $P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*}$ is the program given in Figure 6.

Inputs: ℓ, u, π, t
Constants: $u_i, \mathsf{F}^{u_i}, H^*, h^*, K^*$

1. Run $\mathsf{Ver}(H^*, h^*, \ell, u, \pi)$ to check that π is an opening of h^* at index ℓ to identity u. If the check fails, abort and output $\perp$.
2. Check that $u \neq u_i$. If the check fails, abort and output $\perp$.
3. Check that $\mathsf{G}(t) = \mathsf{G}(\mathsf{F}^{u_i}(u))$. If the check fails, abort and output $\perp$.
4. Output K^*.

Figure 6: The program $P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*}$.

To prove that **Hybrid** $i.1$ is indistinguishable from **Hybrid** $(i-1)$, we need to perform several intermediate hybrids.

Hybrid $i.1.j$. This is identical to **Hybrid** $(i-1)$, except that $\widehat{P_{ciph}}^*$ is replaced with $\mathsf{iO}(P_{ciph}^{u_i,\mathsf{F},H^*,K^*,j})$ where $P_{ciph}^{u_i,\mathsf{F},H^*,h^*,K^*,j}$ is the program given in Figure 7.

Notice that if $j = 0$, the check in Line 2 of Figure 7 never fails, so $P_{ciph}^{u_i,\mathsf{F},H^*,h^*,K^*,0}$ is functionally equivalent to $P_{ciph}^{\mathsf{F},H^*,h^*,K^*}$. Thus **Hybrid** $i.1.0$ is indistinguishable from **Hybrid** $(i-1)$.

Now, if $j = |S^*|$, then the check in Line 2 of Figure 7 *always* fails on input u_i. Therefore, $P_{ciph}^{u_i,\mathsf{F},H^*,h^*,K^*,|S|}$ is functionally equivalent to $P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*}$. Thus **Hybrid** $i.1.|S|$ is indistinguishable from **Hybrid** $i.1$.

Now we wish to show that **Hybrid** $i.1.j$ is indistinguishable from **Hybrid** $i.1.(j-1)$. We do this though additional intermediate hybrids.

Hybrid $i.1.j.1$. This is identical to **Hybrid** $i.1.(j-1)$, except that $H^* \leftarrow \mathsf{Gen}(|S^*|, j)$. By the index hiding property of the hash, this change is undetectable.

Inputs: ℓ, u, π, t
Constants: $u_i, \mathsf{F}, H^*, h^*, K^*, j$

1. Run $\mathsf{Ver}(H^*, h^*, \ell, u, \pi)$ to check that π is an opening of h^* at index ℓ to identity u. If the check fails, abort and output $\perp$.
2. If $u = u_i$, check that $\ell \leq j$. If the check fails, abort and output $\perp$.
3. Check that $\mathsf{G}(t) = \mathsf{G}(\mathsf{F}(u))$. If the check fails, abort and output $\perp$.
4. Output K^*.

Figure 7: The program $P_{ciph}^{u_i,\mathsf{F},H^*,h^*,K^*,j}$.

Hybrid $i.1.j.2$**.** This is identical to **Hybrid** $i.1.j.1$ except that $\widehat{P_{ciph}}^*$ is now set to $\mathsf{iO}(P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*,j})$ (as apposed to $\mathsf{iO}(P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*,j-1})$). Since H^* is statistically binding on index j, and since S^* does is not contain u_i as it's jth identity (since $u_i \notin S^*$), the programs $P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*,j-1}$ and $P_{ciph}^{u_i,\mathsf{F}^{u_i},H^*,h^*,K^*,j}$ are equivalent programs. Thus by the indistinguishability of obfuscations, this change is undetectable.

Finally, notice that **Hybrid** $i.1.j.2$ is identical to **Hybrid** $i.1.j$, except that in the first hybrid, $H^* \leftarrow \mathsf{Gen}(|S^*|, j)$, whereas in the second, $H^* \leftarrow \mathsf{Gen}(|S^*|, 1)$. By the index hiding property of the hash, this change is undetectable.

Thus we have proved that **Hybrid** $(i-1)$ is indistinguishable from **Hybrid** $i.1$. Now notice that in **Hybrid** $i.1$, the entire experiment except for $\mathsf{sk}_{u_i} = \mathsf{F}(u_i)$ can be simulated with F^{u_i}.

Hybrid $i.2$**.** This is identical to **Hybrid** $i.2$, except that sk_{u_i} is replaced with a fresh random element. The security of F shows that this change is undetectable.

Now notice that **Hybrid** i is identical to **Hybrid** $i.2$, except that $\widehat{P_{ciph}}^*$ is converted back to $P_{ciph}^{\mathsf{F},H^*,h^*,K^*}$. This can be proved indistinguishable to **Hybrid** $i.2$ the same way Hybrid $(i-1)$ and $i.1$ where proved indistinguishable. Thus, **Hybrid** $(i-1)$ is indistinguishable from **Hybrid** i.

At this point, we can move to **Hybrid** q_{before} without the adversary noticing, where we have converted all of the secret keys before the challenge ciphertext into random elements. Now we define our target hybrid:

Hybrid *Final***.** This hybrid is identical to **Hybrid** q_{before}, except that $\widehat{P_{ciph}}$ is replaced with an obfuscation of the circuit that outputs $\perp$ on all inputs. Notice that this circuit information-theoretically contains no information about the message encryption key K^*, and therefore the adversary has no advantage in this game.

It remains to show that **Hybrid** *Final* is indistinguishable from **Hybrid** q_{before}. The proof is somewhat similar to the proof that **Hybrid** $(i-1)$ and **Hybrid** i are indistinguishable. We do this through a sequence of hybrids.

Inputs: ℓ, u, π, t
Constants: $j, \mathsf{F}, H^*, h^*, K^*$

1. Run $\mathsf{Ver}(H^*, h^*, \ell, u, \pi)$ to check that π is an opening of h^* at index ℓ to identity u. If the check fails, abort and output $\perp$.
2. If $\ell \le j$, abort and output $\perp$.
3. Check that $\mathsf{G}(t) = \mathsf{G}(\mathsf{F}(u))$. If the check fails, abort and output $\perp$.
4. Output K^*.

Figure 8: The program $P_{ciph}^{j,\mathsf{F},H^*,h^*,K^*}$.

Hybrid $Final.j$**.** for $j = 0, ..., |S|$. In this hybrid, $\widehat{P_{ciph}}$ is replaced by $\mathsf{iO}(P_{ciph}^{j,\mathsf{F},H^*,K^*})$ where $P_{ciph}^{j,\mathsf{F},H^*,h^*,K^*}$ is the program in Figure 8.

Notice that if $j = 0$, then $P_{ciph}^{\mathsf{F},H^*,h^*,K^*,0}$ is equivalent to $P_{ciph}^{\mathsf{F},H^*,h^*,K^*}$, and by IO, we have that **Hybrid** $Final.0$ is indistinguishable from **Hybrid** q_{before}. Also, if $j = |S|$, then $P_{ciph}^{\mathsf{F},H^*,h^*,K^*,|S|}$ is equivalent to the circuit that always outputs $\perp$, and so by IO, we have that **Hybrid** $Final.|S|$ is indistinguishable from **Hybrid** $Final$.

Therefore, it remains to show that **Hybrid** $Final.j$ is indistinguishable from **Hybrid** $Final.(j-1)$. We do this through several more intermediate hybrids.

Hybrid $Final.j.1$**.** This is identical to **Hybrid** $Final.j$, except that $H^* \leftarrow \mathsf{Gen}(|S|, j)$. The index hiding property of H^* shows that this change is undetectable.

Hybrid $Final.j.2$**.** This is identical to **Hybrid** $Final.j.1$, except that $\widehat{P_{ciph}}$ is replaced with $\mathsf{iO}(P_{ciph}^{j,u_j,y,\mathsf{F}^{u_j},H^*,h^*,K^*})$ where u_j is the jth identity in S^*, $y = \mathsf{G}(\mathsf{F}(u_j))$, and $P_{ciph}^{j,u_j,y,\mathsf{F}^{u_j},H^*,h^*,K^*}$ is the program given in Figure 9.

$P_{ciph}^{j,u_j,y,\mathsf{F}^{u_j},H^*,h^*,K^*}$ and $P_{ciph}^{j-1,\mathsf{F},H^*,h^*,K^*}$ only act differently in the $\ell = j$ case. If $u \neq u_j$, then since H^* is statistically binding at position j and S^* has u_j at position j, $P_{ciph}^{j-1,\mathsf{F},H^*,h^*,K^*}$ would have aborted. In the case where $u = u_j$, then $P_{ciph}^{j-1,\mathsf{F},H^*,h^*,K^*}$ would have checked that $\mathsf{G}(t) = \mathsf{G}(\mathsf{F}(u_j)) = y$. Thus, the two programs have identical behavior, and IO shows that **Hybrid** $Final.j.2$ is indistinguishable from **Hybrid** $Final.j.1$.

Write $y = \mathsf{G}(x)$ where $x = \mathsf{F}(u_j)$. Notice that the entire experiment in **Hybrid** $Final.j.2$, except for computing x, can be simulated with F^{u_j}.

Hybrid $Final.j.3$ This hybrid is identical to **Hybrid** $Final.j.2$ except that $y = \mathsf{G}(x)$ for a random x. The security of F shows that this change is undetectable.

Hybrid $Final.j.4$ This hybrid is identical to **Hybrid** $Final.j.3$, except that y is chosen uniformly at random. The security of G shows that this change is undetectable.

At this point, since G is expanding, y is, with high probability, not in the image space of G.

```
Inputs: $\ell, u, \pi, t$
Constants: $j, u_j, y, \mathsf{F}^{u_j}, H^*, h^*, K^*$

  1. Run $\mathsf{Ver}(H^*, h^*, \ell, u, \pi)$ to check that $\pi$ is an opening of $h^*$ at index $\ell$ to identity $u$. If the
     check fails, abort and output $\perp$.
  2. If $\ell \leq j-1$, abort and output $\perp$.
  3. If $\ell = j$, then:
     (a) If $u \neq u_j$, abort and output $\perp$.
     (b) If $u = u_j$, then check that $\mathsf{G}(t) = y$. If not, abort and output $\perp$.
  4. If $\ell > j$, check that $\mathsf{G}(t) = \mathsf{G}(\mathsf{F}(u))$. If the check fails, abort and output $\perp$.
  5. Output $K^*$.
```

Figure 9: The program $P_{ciph}^{j,u_j,y,\mathsf{F}^{u_j},H^*,h^*,K^*}$.

Hybrid *Final.j*.5 This hybrid is identical to **Hybrid** *Final.j*.4, except that $\widehat{P_{ciph}}$ is replaced with $\mathsf{iO}(P_{ciph}^{j,\mathsf{F},H^*,h^*,K^*})$. Since y is not in the image space of G, the program $P_{ciph}^{j,u_j,y,\mathsf{F}^{u_j},H^*,h^*,K^*}$ aborts whenever $\ell = j$, and therefore $P_{ciph}^{j,u_j,y,\mathsf{F}^{u_j},H^*,h^*,K^*}$ and $P_{ciph}^{j,\mathsf{F},H^*,h^*,K^*}$ are equivalent programs. Thus this change is undetectable.

Finally, notice that **Hybrid** *Final.j* is identical to **Hybrid** *Final.j*.5, except that $H^* \leftarrow \mathsf{Gen}(|S|, 1)$. The index hiding of H^* shows that these hybrids are indistinguishable. Thus **Hybrid** *Final.j* is indistinguishable from *Final.*$(j-1)$.

To summarize, we have gradually moved to a situation where the ciphertext contains no information about the message encryption key K^*. All these changes were undetectable to the adversary, and the adversary has advantage zero in the final setting. Therefore, the adversary's advantage in the actual security experiment must have been negligible.

□

4.4 Making Decryption Statistically Independent

In order to use our conversion to CCA security described in Section 5, we need our decryption procedure to have outputs that are statistically independent of the secret key used. We can assume the secret keys are well-formed, but this property must hold for any possible ciphertext, even malformed ones. Since our ciphertexts are obfuscated programs, it is fairly easy for the adversary to create malformed ciphertexts. Therefore, we need a way for the adversary to prove that the ciphertext is generated correctly. It is straightforward to use non-interactive zero knowledge (NIZK) proofs to achieve this goal. Essentially, the key generation procedure also outputs a common reference string for a statistically sound NIZK. The encrypter then proves that his ciphertext is generated correctly. The soundness of the NIZK shows that the ciphertext is correctly generated. Once we guarantee that all ciphertexts are correctly formed, the statistically independent decryption follows from the correctness of the scheme.

5 Boosting to CCA Security

In this section, we show how to transform our schemes into CCA-secure broadcast scheme. More generally, we show how to transform any broadcast scheme with statistically-independent decryption into a CCA-secure broadcast scheme.

Construction 5.1. Let the tuple $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ be a broadcast encryption scheme. Let the tuple $(\mathsf{Gen}, \mathsf{sign}, \mathsf{Ver})$ be a strongly unforgeable one-time signature scheme with w-bit verification keys.

- $\mathsf{Setup}(\mathcal{ID}) = \mathsf{Setup}'(\mathcal{ID}' = \mathcal{ID} \cup \{u_{i,b}\}_{i\in[w],b\in\{0,1\}})$. That is, the identity space of the underlying CPA-secure broadcast scheme consists of "real" identities for the derived scheme, and $2w$ special identities $u_{i,b}$.
- $\mathsf{KeyGen}(\mathsf{msk}, u) = \mathsf{KeyGen}'(\mathsf{msk}, u)$
- $\mathsf{Enc}(\mathsf{Params}, S)$: Run $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}()$. Let $S' = S \cup \{u_{i,\mathsf{vk}_i}\}_{i\in[k]}$. That is, use the bits of vk to select half of the $u_{i,b}$, and add those to S. Run $(\mathsf{Hdr}', K) \leftarrow \mathsf{Enc}'(\mathsf{Params}, S')$ and $\sigma \leftarrow \mathsf{sign}(\mathsf{sk}, (S, \mathsf{Hdr}'))$. Output $(\mathsf{Hdr} = (\mathsf{Hdr}', \mathsf{vk}, \sigma), K)$.
- $\mathsf{Dec}(\mathsf{Params}, u, \mathsf{sk}_u, S, (\mathsf{Hdr}', \mathsf{vk}, \sigma))$: Check that $\mathsf{Ver}(\mathsf{vk}, (S, \mathsf{Hdr}'), \sigma)$ accepts, and if not output $\perp$. Otherwise, use vk to construct S' as above. Then run $K' \leftarrow \mathsf{Dec}'(\mathsf{Params}, u, \mathsf{sk}_u, S', \mathsf{Hdr}')$.

Correctness. The correctness of the scheme follows immediately from the correctness of the underlying scheme and the fact that $S \subseteq S'$.

Parameter Sizes. $2w$ identities are added to the identity space and w recipients are added, where w is related to the security parameter (namely, w is independent of the size of the identity-space $\mathcal{ID}$). For non-identity-based schemes (where $\mathcal{ID}$ is polynomial in size) the number of possible identities is still polynomial, so this conversion can be used with such schemes. Since all system parameters already grow with the security parameter, the cost of this conversion is absorbed into terms involving the security parameter. Thus the asymptotics of the scheme are not significantly affected. Furthermore, when the number of recipients is far larger than the security parameter (which is the interesting domain for broadcast encryption), the overhead for this conversion is minimal.

Security. We now prove the security of our scheme. The idea behind the proof is simple: the reduction will choose $(\mathsf{vk}^*, \mathsf{sk}^*) \leftarrow \mathsf{Gen}()$ for the challenge ciphertext, and obtain the secret keys for users $u_{i,1-\mathsf{vk}_i}$. Now during any decryption query targeting user u, if $\mathsf{vk} \neq \mathsf{vk}^*$, the reduction will be able to decrypt the ciphertext using one of its secret keys. The statistical independence of decryption insures that the result "looks" like it was decrypted using u, even though the reduction may not have the secret key for u. If $\mathsf{vk} = \mathsf{vk}^*$, and the ciphertext is not equal to the challenge ciphertext, then the ciphertext query constitutes a forgery for the signature scheme. Thus, our reduction will be able to either respond to the decryption queries, or break the signature scheme.

Theorem 5.2. *Let* $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ *be an adaptively (resp. semi-statically, statically) CPA secure broadcast scheme with statistically independent decryption. Let* $(\mathsf{Gen}, \mathsf{sign}, \mathsf{Ver})$ *be a strongly one-time secure signature scheme. Then Construction 5.1 is adaptively (resp. semi-statically, statically) CCA secure.*

Proof. We prove the adaptive case, the other cases being similar. Let $\mathcal{A}$ be an adaptive CCA security adversary for Construction 5.1. We prove security though a sequence of games:

Game 0. This corresponds to the real security game. We can assume that $(\mathsf{vk}^*, \mathsf{sk}^*)$ for the challenge ciphertext are chosen at the start of the game. Let $\epsilon = \epsilon_0$ be the advantage of $\mathcal{A}$ in this game.

Game 1. In this game, if the challenger ever receives a decryption query with $\mathsf{vk} = \mathsf{vk}^*$, it outputs a random bit and aborts. Let ϵ_1 be the advantage of $\mathcal{A}$ in **Game 1**. If the adversary is able to query on a ciphertext with $\mathsf{vk} = \mathsf{vk}^*$ that is different from the challenge ciphertext, it must have forged a new signature relative to vk^*. Therefore, such an adversary can be used to break the one-time security of $(\mathsf{Gen}, \mathsf{sign}, \mathsf{Ver})$. The security of $(\mathsf{Gen}, \mathsf{sign}, \mathsf{Ver})$ then shows that ϵ_1 is negligibly close to ϵ_0.

Game 2. In this game, the challenger runs $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}, u_{i,1-\mathsf{vk}_i^*})$ for $i \in [w]$ after generating the scheme parameters. Now, in every decryption query with verification key vk, since $\mathsf{vk} \neq \mathsf{vk}^*$, there is some i such that $\mathsf{vk}_i = 1 - \mathsf{vk}_i^*$. The challenger uses then sk_i to decrypt the underlying header Hdr' instead of the secret key for the user specified by the adversary. Let ϵ_2 be the advantage of $\mathcal{A}$ in **Game 2**. The statistical independence of decryption implies that this change is undetectable to $\mathcal{A}$. Therefore, ϵ_2 is negligibly close to ϵ_1.

Observe that in **Game 2**, the challenger never computes a secret key for users u_{i,vk_i^*} (by the definition of **Game 2**) or users $u \notin S^*$ (since such secret key queries are not allowed in the security game). Moreover, the challenge query yields and encryption to the set $S' = S^* \cup \{u_{i,\mathsf{vk}_i^*}\}_{i \in [w]}$. Since the challenger never computes any secret key for users in S', the challenger and adversary $\mathcal{A}$ can be used to break the CPA security of $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with advantage ϵ_2, which must therefore be negligible. Thus $\epsilon = \epsilon_0$ is negligible as well, as desired. □

Acknowledgments

We would like to thank Daniel Wichs for helpful discussions regarding the used of somewhere statistically binding hash functions. 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.

[ABSV14] Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. The trojan method in functional encryption: From selective to adaptive security, generically. Cryptology ePrint Archive, Report 2014/917, 2014. http://eprint.iacr.org/2014/917.

[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.

[BCHK07] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.

[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.

[BWZ14a] 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.

[BWZ14b] Dan Boneh, David J. Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. Cryptology ePrint Archive, Report 2014/930, 2014. http://eprint.iacr.org/2014/930.

[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.

[CHL$^+$14] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. Cryptology ePrint Archive, Report 2014/906, 2014. http://eprint.iacr.org/2014/906.

[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.

[CLT14] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, 2014. http://eprint.iacr.org/2014/975.

[CLT15] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. Cryptology ePrint Archive, Report 2015/162, 2015. http://eprint.iacr.org/.

[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. Graph-induced 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.

[GGHZ14a] 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.

[GGHZ14b] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666, 2014. http://eprint.iacr.org/2014/666.

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

[GHMS14] Craig Gentry, Shai Halevi, Hemanta K. Maji, and Amit Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. Cryptology ePrint Archive, Report 2014/929, 2014. http://eprint.iacr.org/2014/929.

[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.

[HW14] Pavel Hubacek and Daniel Wichs. On the communication complexity of secure function evaluation with long output. Cryptology ePrint Archive, Report 2014/669, 2014. http://eprint.iacr.org/2014/669.

[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.

[PPSS12] Duong Hieu Phan, David Pointcheval, Siamak Fayyaz Shahandashti, and Mario Strefler. Adaptive CCA broadcast encryption with constant-size secret keys and ciphertexts. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *ACISP 12: 17th Australasian Conference on Information Security and Privacy*, volume 7372 of *Lecture Notes in Computer Science*, pages 308–321, Wollongong, NSW, Australia, July 9–11, 2012. 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/2014/588.

[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.