Limits on Adaptive Security for Attribute-Based Encryption

Zvika Brakerski* Stav Medina*

Abstract

This work addresses the long quest for proving full (adaptive) security for attribute-based encryption (ABE). We show that in order to prove full security in a black-box manner, the scheme must be "irregular" in the sense that it is impossible to "validate" secret keys to ascertain consistent decryption of ciphertexts. This extends a result of Lewko and Waters (Eurocrypt 2014) that was only applicable to straight-line proofs (without rewinding). Our work, therefore, establishes that it is impossible to circumvent the irregularity property using creative proof techniques, so long as the adversary is used in a black-box manner.

As a consequence, our work provides an explanation as to why some lattice-based ABE schemes cannot be proven fully secure, even though no known adaptive attacks exist.

1 Introduction

An Attribute-Based Encryption scheme (ABE) [SW05, GPSW06] is one that allows fine-grained access to encrypted data by issuing multiple secret keys, each with its own permissions, and protecting the privacy of ciphertext even against colluding unauthorized parties. More explicitly, an ABE scheme consists of a global pair of "master" public-key (also known as the public parameters of the scheme) and secret-key, where the former is used to encrypt messages, and the latter is used to issue individual decryption keys. Messages are encrypted, using the public parameters, with respect to an *attribute* x (for our purposes $x \in \{0,1\}^n$). Secret keys are generated using the master secret-key, with respect to predicate functions f, so that SK_f can decrypt all ciphertexts with attributes x for which $f(x) = 1$.[1] The security requirement is *collusion resilience*. Namely, even if an attacker has as many SK_{f_i} as they want, if $f_i(x) = 0$ for all i, then the attacker cannot decrypt ciphertexts with attribute x. ABE schemes that support sufficiently rich function classes (even the class of shallow circuits or the class of boolean formulae) are known to exist only under two types of cryptographic assumptions: assumptions on groups with bilinear maps [SW05, GPSW06, OSW07, Wat11, LOS+10, LW12, KL15, CGKW18, KW20, GW20, LL20] and lattice assumptions [AFV11, ABV+12, Boy13, GVW13, BGG+14].

ABE proved to be a useful primitive for many purposes (see [LOS+10, AFV11, DDM15, GKW17, DGP21] for just a few samples among many). However, proving security for ABE is a challenging task. In the security reduction, the ABE adversary is used to violate the underlying hardness assumption. Since the ABE adversary is allowed to request multiple keys SK_{f_i}, the proof needs to be designed so that such keys can be generated and fed to the adversary, where at the same time, there is a challenge of the underlying assumption that remains unsolved in the eyes of the reduction algorithm. Therefore, in many cases, security is proved in a relaxed model, which is known as *selective security*. In this model, the adversary needs to declare, ahead

*Weizmann Institute of Science, Israel, `{zvika.brakerski,stav.medina}@weizmann.ac.il`. Supported by the Israel Science Foundation (Grant No. 3426/21), and by the European Union Horizon 2020 Research and Innovation Program via ERC Project REACT (Grant 756482).

[1] The description here is for a variant known as Key-Policy ABE (KP-ABE). There is another variant known as Ciphertext-Policy ABE (CP-ABE) where encryption is with respect to f, and secret-key generation is with respect to x. The distinction is not very important for our purposes, so we adopt the KP-ABE notation throughout this manuscript.

of time, the value x^* on which it wishes to violate security [CHK03]. This allows the reduction to design the public parameters so that it is possible to generate SK_{f_i} for which $f_i(x^*) = 0$. However, this naturally restricts the attacker's power since, in an actual attack, the adversary may be exposed to some SK_{f_i} and only then choose x^*. Protecting against the latter is known as full security, or as *adaptive security*, to emphasize the possibility of the aforementioned adaptive attacks. An intermediate notion, semi-adaptive security, allows x^* to be chosen after seeing the public parameters but before seeing any actual key. This latter notion appears to be more similar to selective than to adaptive security and can be achieved in similar ways to the selective case [BV16, GKW16].

Adaptive security is notoriously hard to prove. Intuitively, this is because the reduction needs to be prepared to "feed" the adversary with a key of their choice. For example, the reduction does not know whether the adversary will ask for a key for a function f or for its complement. Therefore, in a sense, the reduction should have the ability to decrypt any ciphertext without the adversary's help. Indeed, until recently, fully secure ABE was *only* known to exist under assumptions on bilinear maps, mostly using the dual-system technique of Waters and its successors [Wat09]. [2] Recently, Tsabary [Tsa19] showed an approach towards full security in the lattice regime, but only for a very restricted class of functions f (in particular, this is still not known for general shallow circuits or for the class of boolean formulae).

Nevertheless, as hard as it is to prove adaptive security, actual adaptive attacks are not so common. In fact, we are not aware of adaptive attacks against the lattice-based schemes of [GVW13, BGG$^+$14]. It is, therefore, quite puzzling that we need to apply involved techniques and lose a lot of functionality (as of yet) in order to prove this property.

Lewko and Waters [LW14] tried to formalize the above intuition on the hardness of proving adaptive security. They noticed that since the reduction needs to be able to produce keys that essentially violate the security of any individual ciphertext, one can *extract from the reduction itself* information that allows violating the hardness assumption, thus trivializing the proof. This extraction is done via *rewinding*. At a high level, we consider an algorithm that runs the reduction as an adversary, asks for a key for the function f_0, and then *rewinds* the reduction to the point before the key was queried. It then asks for a different key f_1 and claims to violate security for some x^* for which $f_0(x^*) = 1$ and $f_1(x^*) = 0$. Indeed, security is violated by using SK_{f_0} that was obtained in the rewound part of the execution. The current thread of the reduction "thinks" that only f_1 was queried, thereby assuming it is interacting with a legitimate, successful ABE adversary. This should lead to the reduction of violating the hardness assumption in polynomial time.

One has to be careful when applying this argument (especially given that adaptive security via black-box reductions *is* possible to achieve in some cases). In order for it to work, the reduction should not notice that the challenge ciphertext is being decrypted by a key that was obtained in another thread. Therefore, SK_{f_0} should decrypt ciphertexts in a "canonical" manner that does not expose the origins of the key. Lewko and Waters, therefore, defined a criterion for secret keys and ciphertexts, essentially requiring that it is not possible to distinguish different decryptions of a ciphertext, even if they were obtained using different keys (so long as all secret keys and ciphertext involved pass a public validation procedure). This is a very natural property, and thus the Lewko-Waters result has the following very strong implication. In order to achieve provable adaptive security, one must forgo the ability to validate secret keys and ciphertexts to ensure that decryption is done in a consistent manner. We are not aware of a setting where this "checkability" property is explicitly required, but we imagine that the ability to validate keys and ciphertexts may be desirable in a multi-user system.

Existing methods for constructing fully secure ABE do this by making their scheme impossi-

[2] We note that there is a way to generically upgrade selective to adaptive security, at the cost of a 2^n factor degradation in security, by simply guessing the value of x^*. This is known as *complexity leveraging*. However, the cost is prohibitive in many cases, and furthermore, this method cannot be applied if n is not a-priori bounded.

ble to validate. This applies to Waters's aforementioned dual-system technique, where the proof utilizes "semi-functional" keys and ciphertexts, which differ in functionality from "regular" keys, thus inherently violating the checkability property. Tsabary's approach relies on generating a special ciphertext that some policy-accepting honestly-generated secret keys cannot decrypt. Therefore, in the security proof, the challenge ciphertext would decrypt to different values if the attacker attempted to decrypt it with different (policy-accepting) secret keys. Here again, it is inherently impossible to come up with a validation procedure for the scheme. We note that one can also derive adaptively-secure ABE from a stronger notion known as (adaptively-secure) Functional Encryption (FE) which is intimately related to program obfuscation [GGH+13]. Known constructions of adaptively-secure FE [Wat15, ABSV15] use a so-called "punctured programs" technique which, similarly to dual-systems inherently violate checkability.

An important limitation of the Lewko-Waters result is that it does not apply when the reduction itself rewinds the adversary. Indeed, rewinding is a very common proof technique in cryptography. For example, the reduction, upon receiving a request to provide a key for f_0, may rewind the adversary and test it on public parameters that the reduction generated by itself in order to see that it actually manages to solve ABE "dummy challenges" before actually providing it with keys with respect to the "real" public parameters.

If we then try to apply the outline above, our algorithm tries to rewind the reduction, but then the reduction, in turn, attempts to rewind the attacker. Our algorithm, therefore, needs to pretend to have been rewound and solve the dummy challenges, which again requires rewinding the reduction. Rewinding seems to complicate the above outline significantly, and indeed, Lewko and Waters only applied their techniques to straight-line reductions – ones that do not use rewinding. This still leaves hope that maybe, if we are clever enough about proof techniques, we can come up with a fully secure ABE scheme that is as simple as our existing selective schemes. In fact, perhaps it is even possible to prove adaptive security for [GVW13, BGG+14] using a sufficiently clever reduction.

1.1 Our Results

Our main result is to extend the result of Lewko and Waters to handle rewinding reductions. Along the way, we introduce a simpler notion of *checkability*. Therefore, our result shows that the Lewko-Waters argument cannot be circumvented by clever proof techniques (so long as the adversary is used in a black-box manner), and necessarily, there is a cost for the ability to prove adaptive security. Both in terms of naturalness and possibly performance.

Whereas Lewko and Waters considered security proofs that consisted of a simple black-box reduction to a non-interactive hardness assumption, we allow the underlying hardness assumption to be interactive and merely require that the number of communication rounds is a priori bounded by a fixed polynomial. This is a natural and essentially necessary requirement since otherwise we could take the adaptive security of the scheme to be the underlying hardness assumption, and trivially the adaptive security reduces to itself.

As an implication of our main theorem, we show that (the delegatable version of)[3] the celebrated lattice-based scheme of [BGG+14] cannot be proven adaptively secure, at least without modifications. To this end, we observe that the security definition of an ABE scheme does not involve the decryption algorithm. Indeed, an attacker receives keys and a challenge ciphertext and attempts to recover the message that has been encrypted. The entire security game is conducted without any party being "instructed" to use the decryption algorithm.[4] We show that these schemes have an alternative decryption algorithm and that, with respect to this decryption algorithm, it is possible to validate secret keys and ciphertexts. Therefore, our result can be applied to rule out adaptive security reductions for this scheme. An important takeaway here

[3]In the delegatable version, the secret key contains a lattice trapdoor rather than a single vector. See Section 1.2 for further explanation and discussion on the implications for the non-delegatable version.

[4]This property is not unique to ABE, and it also occurs in standard CPA secure encryption.

is that in order to rule out validation, one must consider *all possible decryption circuits* (that decrypt correctly) and effectively show that it is impossible to validate secret keys and ciphertexts with respect to all of them. We then discuss some modifications to the aforementioned scheme and their impact on the ability to validate. Our conclusion is that apparently a more radical change, as per [Tsa19], may be required in order to be able to prove adaptive security in the lattice setting.

1.2 Technical Overview

To prove our main theorem, we consider a Turing reduction $\mathcal{R}$ from some intractability assumption $\mathcal{C}$ to violating the adaptive security of some ABE scheme. The assumption $\mathcal{C}$ is stated in terms of an interactive game (following the framework of [Nao03]). The reduction asserts that if there existed a successful ABE adversary $\mathcal{A}$, then $\mathcal{R}$ could use it in a black-box manner in order to break the assumption $\mathcal{C}$. Following [LW14], the underlying idea of our proof is to use the reduction $\mathcal{R}$ directly in order to break the assumption $\mathcal{C}$, without the help of an actual ABE attacker. In order to do so, we wish to efficiently emulate an ABE attacker for $\mathcal{R}$, in a way that $\mathcal{R}$ would not be able to distinguish from a real attacker.

Adaptive security is defined by an interactive game between the reduction $\mathcal{R}$, i.e. the challenger, and the attacker $\mathcal{A}$. The game begins with the challenger declaring the public parameters of the ABE scheme. Then, the attacker can make key queries to the challenger for secret keys of predicates of its choice. Next, it declares a challenge attribute x^*, and receives from the challenger a challenge ciphertext CT_{x^*} encrypted w.r.t. x^* (say, encrypting either the message 0 or 1). Afterward, $\mathcal{A}$ can make more key queries until finally, it sends a guess of which message was encrypted in CT_{x^*}. The adversary wins the game if it has not received any secret key that accepts x^* and the guess has been correct. We say that an adversary breaks the security assumption if it wins the game with a noticeable advantage compared to randomly guessing.

Recall that in order to emulate an attacker for $\mathcal{R}$, our algorithm should be able to solve possible "dummy challenges" and properly decrypt ciphertexts provided by the reduction. Naively, this could be achieved by rewinding the reduction and extracting additional secret keys. However, as we explained above, two issues arise: (1) First, the reduction may notice when the simulated attacker decrypts the challenge ciphertext using a secret key that was extracted by rewinding the reduction, and then behave unexpectedly as a result, so we would not be able to use the reduction to violate $\mathcal{C}$. (2) Second, once we allow the reduction to rewind the attacker, the naive strategy could lead to many "nested" rewindings of the reduction and the attacker, which in turn may result in an exponential running time. Furthermore, when considering a general assumption $\mathcal{C}$, we have to ensure that rewinding the reduction does not affect the interaction between $\mathcal{R}$ and $\mathcal{C}$, since $\mathcal{C}$ itself is not in our control and we are unable to rewind it.

In [LW14], Lewko and Waters addressed the first obstacle and defined a *checkability* criterion for ABE schemes, which states that each secret key and each ciphertext can be validated to ensure "canonical" decryption. Namely, the checkability property requires that a valid ciphertext is decrypted to the same message using any valid secret key. This way, the simulated attacker can validate the challenge ciphertext and the secret key used for decryption, and the reduction would not detect which secret key was used. We present a simpler checkability requirement, which states that each secret key can be validated and that the decryption of ciphertext using any valid secret key results in the same decryption distribution. This property is sufficient to ensure consistent decryption by applying the same attacker strategy. Note that every scheme that satisfies Lewko and Waters's checkability requirement can be trivially transformed to satisfy our checkability requirement by validating the ciphertext at the beginning of the decryption procedure and outputting $\perp$ if found invalid. We further observe that the implementation of the decryption procedure of an ABE scheme does not play any role in the adaptive security proof. Therefore, if we modified the decryption procedure of an ABE scheme so it would satisfy our checkability property and show it cannot be proved adaptively secure, then we would conclude that the original scheme could not be proven adaptively secure as well.

To overcome the second obstacle, we adopt the more delicate rewinding technique introduced by Pass [Pas11] in the context of witness-hiding special-sound protocols for unique relations. Pass used the rewinding technique in order to obtain a sufficient amount of interaction transcripts, which are essentially proofs of some statement x with different suffixes. The transcripts are then fed to the special-soundness extractor in order to recover a witness for x. We employ similar tools in order to rewind an ABE challenger, obtain multiple keys with respect to the same public parameters, and use them for the purpose of decrypting the challenge ciphertext.

Adopting Pass's terminology, we define the notion of a *slot*, a "time window" during the execution of $\mathcal{R}$. This window "opens" when the emulated ABE attacker makes a key query to obtain a secret key for some predicate function, and it "closes" right after the reduction responds with one and the attacker validates it. Intuitively, this is the shortest time frame that could be rewound in order to extract additional validated secret keys from the reduction. As previously described, due to the recursive nature of rewinding both the reduction and the simulated attacker, the reduction may rewind the attacker at some point during a slot, so another "nested" slot may open. We consider a slot to be "good" for rewinding if, between the time that it opens and the time that it closes, there is no communication between $\mathcal{R}$ and $\mathcal{C}$, and in addition, $\mathcal{R}$ does not rewind the attacker "too many times" within the slot. We also require the extracted key to be a valid one. Intuitively, the first condition ensures that rewinding the slot does not disturb the interaction between $\mathcal{R}$ and $\mathcal{C}$, and the second condition allows us to bound the recursion depth and limit the growth in runtime due to possible nested slots being rewound. Once we encounter a good slot in the execution, we rewind it several times to extract multiple secret keys. The precise criterion for a slot being good is determined with respect to the maximal running time of the reduction and the recursive depth of the slot. This, combined with the fact that $\mathcal{R}$ is an efficient algorithm and cannot make "too many" queries to the attacker, allows us to ensure the existence of good slots while maintaining the bound on the running time of the emulation. Note that an ABE attacker may query the challenger (embodied by the reduction in our case) many times. It suffices for us that just a fraction of the slots induced by these queries is good because once a slot is good, we can rewind it many times and obtain many secret keys that will allow us to decrypt the challenge ciphertext.

Being able to rewind the reduction is a necessary step toward simulating an attacker, but we must also argue that the rewound reduction will indeed provide us with a valid secret key that will decrypt x^*. To this end, we need to design the function class for which we make queries, as well as the challenge attribute x^*. We aim for these values to meet the following conditions: On the one hand, the secret keys queried during the "mainline" execution of $\mathcal{R}$ should not accept the selected challenge attribute x^*. On the other hand, to successfully decrypt the challenge, at least one of the keys from the rewound execution should accept x^*. Finally, we must meet these two conditions in a way such that the reduction would not be able to detect when it is being rewound; otherwise, it may abort.

We approach this challenge as follows: The challenge attribute x^* is sampled uniformly at random from the set of possible attributes, and each secret key is sampled randomly from a specific set of functions, exactly the same way during the rewound queries as during the "mainline" ones. The set of functions is constructed from a pairwise independent hash family, so we expect the fraction of any subset of attributes covered by a uniformly random function from the set to be close to its expectation (a property also known as mixing). This is a key ingredient in our analysis to ensure any attribute has a high enough probability of being covered by the secret keys we extract by rewinding (this is true even when the reduction may choose not to disclose a fraction of the keys, as we explain shortly). The probability of a function in the set covering some attribute is chosen to be small enough so that with high enough probability, the uniformly random challenge x^* is not covered by the functions queried during the "mainline" execution of $\mathcal{R}$, but large enough so that with high enough probability, x^* is covered by at least one key that was extracted from a rewound execution. By specifying the functions to be queried, we require the ABE scheme to support them as predicates. Fortunately, since pairwise hash families can be implemented by NC^1 circuits [IKOS08], the requirement is commonly satisfied.

We must pay delicate care when computing the probability of x^* being covered by a secret key that was extracted by rewinding since we are not allowed to make any presumptions regarding the behavior of the reduction. In particular, $\mathcal{R}$ may refuse to provide secret keys for certain queries in an unexpected way. However, since $\mathcal{R}$ is a reduction from some hardness assumption $\mathcal{C}$ to an attacker, it should be impossible for $\mathcal{R}$ to violate the assumption $\mathcal{C}$ without the help of the attacker. In other words, the reduction must rely on the attacker breaking adaptive security, so it must cooperate and "play" according to the security game protocol with a high enough probability. Otherwise, it would fail to violate $\mathcal{C}$ with a noticeable advantage. For the same reason, if the reduction would only accept a negligible portion of the possible challenge attributes, it would fail to successfully violate $\mathcal{C}$ with a noticeable advantage. This behavior is a generalization of a method known as *complexity leveraging*, in which the challenger guesses the challenge attribute x^* in advance and rejects any other challenge. The method is used to upgrade a selectively secure scheme to be adaptively secure generically, but at the cost of exponential degradation in security, which, as explained, makes it irrelevant to our scenario.

Implications to Lattice-Based ABE Schemes. We now turn our attention to applying our result to the lattice-based ABE scheme of Boneh et al. [BGG$^+$14]. This scheme supports policy functions represented by boolean circuits of a polynomially a-priori bounded depth (the depth bound is a parameter in the initialization of the scheme), and security is based on the hardness of the Learning with Errors problem (LWE) [Reg05].

We start by outlining the high-level structure of the scheme. In particular, we consider the version that supports key delegation (see discussion on other variants and on other lattice-based ABE schemes at the end of this outline). The public parameters are designed so that each input attribute x is associated with a lattice A_x (we do not define what a lattice is since this is a high-level overview, but we will explain all properties that are relevant to our outline). This lattice can be publicly derived from the public parameters of the scheme. A ciphertext with respect to x consists of a noisy vector that is close to the lattice A_x. More explicitly, A_x is represented as a matrix in $\mathbb{Z}_q^{n\times m}$, and the ciphertext consists of a vector of the form $c = sA_x + e \pmod q$, where s is a uniform vector in $\mathbb{Z}_q^n$, and the noise e is sampled from a distribution over short vectors. The message to be encrypted is then encoded by adding an "offset" to c, which depends on the message. Importantly, being able to recover e from c suffices in order to recover the message m, so the exact encoding procedure is immaterial for our purposes. In terms of key generation, every possible predicate function f is also associated with a (public) lattice A_f. In this version of [BGG$^+$14], the secret key for the predicate f consists of a *trapdoor* for the lattice A_f. A lattice trapdoor can take many forms (that are interchangeable). The simplest one, perhaps, is a "short basis" for the so-called dual lattice. The details are immaterial, but the important property is that given a trapdoor T_A for a lattice A and given a vector $c = sA + e \pmod q$, where $\|e\| \le B$ (for some parameter B that relates to the "quality" of the trapdoor), it is possible to recover e (and furthermore this e is unique). We refer to this operation as *decoding* (due to the similarity of decoding in error correcting codes). Thus, secret keys allow decoding with respect to A_f, but ciphertexts are encoded with respect to different lattices A_x.

The ingenious component of the scheme is a mechanism that allows, for every f, x s.t. $f(x) = 1$, to come up with a low-norm matrix $H = H_{f,x}$ s.t. $A_f = A_x H_{f,x}$ (we do not define what we mean by a norm of a matrix, one can think about its spectral norm, for example). This means that given $c = sA_x + e \pmod q$, it is possible to multiply by H and obtain $c' = sA_f + e' \pmod q$, where e' has a possibly higher norm than e, but the degradation of the norm is bounded and respective to the norm of H. This new c' can be decoded using T_{A_f} in order to decrypt ciphertexts. (We did not explain why recovering e' suffices for decryption, but this can be done.)

To apply our theorem to this scheme, we require the following observation, which follows from the analysis of trapdoor properties in the literature, e.g. [MP12]. It can be shown that $H_{f,x}$ can also be used to translate a trapdoor for A_f into a trapdoor for A_x, with somewhat lower quality.[5] This means that an alternative decryption procedure would be to deduce the trapdoor for A_x and then decode the vector c directly.

For our purposes, we wish to ensure that two different keys, with respect to f_1 or f_2, will decrypt all ciphertext in the same way. We, therefore, take the following strategy. Given a secret key for a function f, we first check that its "quality" as a trapdoor T_{A_f} for A_f is good enough. That is, the honest key generation is guaranteed to produce trapdoors of a certain quality, and when we are given a candidate trapdoor, we check that it is indeed of this quality. Then, given a vector c, we derive a trapdoor T_{A_x} for A_x, and use it to decode c. This is the end of the "standard" decryption procedure, but our "validated" decryption has additional steps. Note that it is quite possible that c is not a legitimate ciphertext at all, and perhaps the outcome e that we obtained is "garbage" and does not actually describe the difference from a nearby lattice point. In such a case, different trapdoors could lead to different "garbage" which contradicts the consistency property we are interested in. Therefore, once we recover the noise vector e, we check whether $(c - e)$ is indeed of the form sA_x (i.e., is a vector in the lattice A_x) and also that $\|e\|$ is sufficiently short (in this case, that its norm is consistent with the norm of a noise vector that is selected by the honest encryption procedure). If either check fails, we return $\perp$, otherwise, we return e (or rather, the message m that is induced by the value of e). The important observation is that any trapdoor for A_x that has been derived by taking a trapdoor for some A_f of the prescribed quality and converting it into a trapdoor for A_x using $H_{f,x}$ should be able to decode "legal" values of e. Therefore, if we get c with an "illegal" value of e, then we always output $\perp$, and if the value of e is "legal", then it should be correctly decoded.

This concludes our alternative decryption algorithm for the [BGG+14] scheme, for which every validated secret key will produce the same output on any given input ciphertext c. Therefore, by our main result, this scheme cannot have a black-box proof of security.

If we wish to construct an adaptively secure scheme, we need to eliminate the checkability property. One direction that comes to mind is to degenerate the [BGG+14] scheme so that it is no longer possible to obtain a trapdoor for A_x. We could try, for example, to change the secret keys of the scheme so that they no longer contain a trapdoor for A_f. Indeed, such a modification is possible, where the secret key for f is modified to only contain a single short vector from the dual lattice rather than a basis. The non-delegatable version of the [BGG+14] scheme indeed works in this way. The predecessor of the [BGG+14] scheme, namely the [GVW13] scheme, also has a similar structure to the non-delegatable [BGG+14], where the secret key does not allow obtaining a full trapdoor for A_x, but only a partial trapdoor (a number of short vectors in the dual lattice, that do not form a basis).

In order for this approach to indeed allow an adaptive security proof, the scheme needs to have properties that seem quite implausible. In particular, it would still be possible to apply our result to obtain a barrier if, instead of querying just one f, the attacker can query many different f's that all accept the same value x. This would allow obtaining a large number of short vectors in the dual lattice of A_x, thus obtaining a trapdoor for x, which would allow for canonical decryption as described above. Due to the convoluted structure of $H_{f,x}$, it is hard to prove that a full basis will be generated in this way, but it seems very implausible that this will not be the case. The situation that we are considering is querying the key generation process on multiple functions f and finding out that on all of the x's that we consider (except a negligible fraction), if we take the dual-lattice vectors for A_x that are generated by all f's that accept x, it holds that they all fall into a proper subspace and do not form a full rank set. Note that

[5] This is a general property. If $A = BH$, and H is short, then given H, a trapdoor for A implies a trapdoor for B, with the "quality" of the trapdoor degrading respective to the norm of H.

the degrees of freedom of the key generation are fairly limited, and the number of x's we can consider is far greater than the number of vectors produced by the key generation. Still, coming up with proof is non-trivial, and we leave it as an open problem. Nevertheless, any attempt to prove adaptive security would require proving the opposite, which seems quite difficult.

Our conclusion, therefore (and others may draw their own), is that one has to deviate from the known methods in order to achieve adaptive security. Indeed, Tsabary's construction [Tsa19] achieves this, for a very limited function class, by significantly deviating from the above blueprint so as to make it impossible to validate the decryption process.

1.3 Paper Organization

Some preliminaries and notation, along with ABE-related definitions are provided in Section 2. Our lower bound argument appears in Section 3. The technical details of our lattice instantiation, which are outlined in the overview above, are provided Section 4.

2 Preliminaries

2.1 Basic Definitions

Let n be a natural number. We denote by 1^n the unary expansion of n, that is, the concatenation of n 1's. We also denote $[n] \stackrel{\text{def}}{=} \{1, \ldots, n\}$.

Definition 2.1. (Negligible Function.) A function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be negligible if for all c there exists N such that $f(n) < n^{-c}$ for all $n > N$. We denote by $\mathrm{negl}(\cdot)$ a negligible function.

Definition 2.2. (Computational Indistinguishability.) Let $\{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{Y_\lambda\}_{\lambda\in\mathbb{N}}$ be two distribution ensembles. We say they are computationally indistinguishable if for any probabilistic polynomial-time algorithm A, it holds that

$$\left| \Pr_{x\leftarrow X_\lambda}[A(x) = 1] - \Pr_{x\leftarrow Y_\lambda}[A(x) = 1] \right| = \mathrm{negl}(\lambda) \tag{1}$$

Definition 2.3. (Statistical Distance.) Let X and Y be two random variables over a finite domain Ω, we define their statistical distance by

$$D(X, Y) \stackrel{\text{def}}{=} \frac{1}{2} \sum_{\omega\in\Omega} |\Pr[X = \omega] - \Pr[Y = \omega]| \tag{2}$$

Definition 2.4. (Statistical Indistinguishability.) Let $\{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{Y_\lambda\}_{\lambda\in\mathbb{N}}$ be two distribution ensembles over a finite domain Ω. We say they are statistically indistinguishable if $D(X_\lambda, Y_\lambda) = \mathrm{negl}(\lambda)$.

Definition 2.5. (Pairwise-Independent Hash Functions.) A family H of functions $h : [n] \to [m]$ is called pairwise independent if for every $i, j \in [n]$ such that $i \neq j$ and every $x, y \in [m]$ it holds that

$$\Pr_{h \xleftarrow{\$} H}[h(i) = x \wedge h(j) = y] = \frac{1}{m^2} \tag{3}$$

2.2 Algorithms

The following algorithms have various alternative definitions. We present the definitions relevant to our analysis and results here and assume familiarity with Turing machines.

Definition 2.6. (Probabilistic Algorithm.) A probabilistic algorithm is a Turing machine that receives an auxiliary random tape as input.

Definition 2.7. (Interactive Algorithm and Interactive Protocol.) An interactive algorithm, or interactive machine, is a Turing machine that has two additional communication tapes, a read-only one and a write-only one. An interactive protocol consists of two interactive machines $\pi = (A, B)$ such that A's write-only communication tape is B's read-only communication tape, and A's read-only communication tape is B's write-only communication tape. The machines take turns (also called "rounds") in being active, each turn ends with the active machine either halting or sending a message to the other machine.

Given a pair of interactive algorithms A and B that interact according to some interactive protocol on some common input x, we denote by $\langle A, B\rangle(x)$ the distribution of the output of B after interacting with A on the common input.

Definition 2.8. (Oracle Machine.) An oracle machine is a Turing machine with access to another machine, called the oracle. The access is implemented using an additional tape, called oracle tape, and two special states, ASK and RESPONSE. The oracle machine may enter the ASK state, which invokes an execution of an oracle on the input received through the oracle tape. The contents of the oracle tape are then replaced with the output of the oracle, and the state is changed to RESPONSE. Let A be an oracle machine and B an oracle; we denote A^B the execution of A with oracle access to B.

Remark 2.9. (Relation between Oracle Machines and Interactive Machines.) Throughout the paper, we treat interactive machines and oracle machines similarly, according to the following equivalency: Consider an interactive protocol (A, B). The execution of the machines interactively is equivalent to executing A as an oracle machine with oracle access to a stateless version of machine B (i.e., machine B without a state register), where every oracle query contains the partial transcript of the interactive protocol, and the output of the oracle is B's next message according to the interactive protocol. Similarly, given an oracle machine A with oracle access to B, we can consider an interactive protocol (A, B) that contains the oracle queries made by A to B and the corresponding responses.

Remark 2.10. (Rewinding an Oracle.) Consider an interactive protocol (A, B) and the corresponding execution of oracle machine A with oracle access to B. Machine A can "rewind" B by making a query to B with input that is a strict prefix of the partial transcript of the interactive protocol, i.e., "rewind" B to a previous state in the interaction.

Definition 2.11. (Decision Problem.) A decision problem is a problem that is solved by classifying inputs to an output which is either 0 or 1.

Definition 2.12. (Black-box Reduction.) A black-box reduction from a decision problem A to a decision problem B is a Turing machine that solves problem A given oracle access to a machine that solves problem B.

2.3 Intractability Assumption

In the following definition, we formally describe the notion of intractability assumption to model a problem that is assumed to be "hard to solve". The assumption $\mathcal{C}$ is stated in terms of an interactive game between a challenger and an adversary (following the framework of [Nao03]).

Definition 2.13. ($r(\cdot)$-round Intractability Assumption with Threshold $t(\cdot)$.) An $r(\cdot)$-round intractability assumption with threshold $t(\cdot)$ is an interactive probabilistic decision problem $\mathcal{C}$, called the challenger, that interacts with another algorithm $\mathcal{A}$, called the attacker, such that: (1) both algorithms take as input 1^λ where λ is the security parameter, and (2) the interaction is a-priori bounded by $r(\lambda)$ rounds. We define the advantage of the attacker $\mathcal{A}$ with respect to $\mathcal{C}$ as

$$\mathrm{Adv}(\mathcal{A}) \stackrel{\text{def}}{=} \left|\Pr\left[\langle \mathcal{A}, \mathcal{C}\rangle\left(1^\lambda\right) = 1\right] - t(\lambda)\right| \tag{4}$$

$\mathcal{C}$ is associated with a computational assumption that states that for any polynomial-time attacker $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that for all $\lambda \in \mathbb{N}$

$$\mathrm{Adv}(\mathcal{A}) \leq \mu(\lambda) \tag{5}$$

We say that a polynomial-time attacker $\mathcal{A}$ breaks the assumption $\mathcal{C}$ with non-negligible advantage p if $\mathrm{Adv}(\mathcal{A}) = p$.

2.4 Attribute-Based Encryption

Definition 2.14. (Key-Policy Attribute-Based Encryption Scheme.) Let $\mathcal{X}$ be a set of objects and $\mathcal{F}$ be a class of functions of the form $f : \mathcal{X} \to \{0,1\}$. A key-policy Attribute Encryption (KP-ABE) scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$ is a tuple of probabilistic polynomial-time algorithms $\mathcal{S} =$ (Setup, Encrypt, KeyGen, Decrypt) as follows:

- **Setup**$(1^\lambda) \to$ PP, MSK   The setup algorithm takes the security parameter λ as input. It outputs the public parameters PP of the scheme and a master secret key MSK.
- **Encrypt**$(x, M, \mathrm{PP}) \to \mathrm{CT}_x$ The encryption algorithm takes in an attribute $x \in \mathcal{X}$, a message $M \in \{0,1\}$ and public parameters PP. It outputs a ciphertext CT_x, which is an encryption of M under x. Assume w.l.o.g. that CT_x contains x.
- **KeyGen**$(\mathrm{MSK}, f, \mathrm{PP}) \to \mathrm{SK}_f$ The key generation algorithm takes in the master secret key MSK, a policy $f \in \mathcal{F}$ and public parameters PP. It outputs a secret key SK_f for f. Assume w.l.o.g. that SK_f contains f.
- **Decrypt**$(\mathrm{CT}_x, \mathrm{SK}_f, \mathrm{PP}) \to$ M   The decryption algorithm takes in a ciphertext CT_x, a secret SK_f and public parameters PP. It outputs a message $M \in \{0,1\}$.

Remark 2.15. Another variant of ABE is Ciphertext-Policy ABE (CP-ABE), where ciphertexts are associated with policies and secret keys with attributes. The distinction is immaterial for our purposes, so we adopt the notation of KP-ABE scheme and simply refer to it as ABE for convenience.

Definition 2.16. ((Perfect) Correctness of KP-ABE.) Let $\mathcal{S} =$ (Setup, Encrypt, KeyGen, Decrypt) be a key-policy ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ is (perfectly) correct if the following holds: Let (PP, MSK) = Setup(1^λ), $f \in \mathcal{F}$, $x \in \mathcal{X}$ and $M \in \{0,1\}$ and suppose $f(x) = 1$. Denote $\mathrm{CT}_x = \mathrm{Encrypt}(x, M, \mathrm{PP})$ and $\mathrm{SK}_f = \mathrm{KeyGen}(\mathrm{MSK}, f, \mathrm{PP})$. It holds that

$$\mathrm{Decrypt}(\mathrm{CT}_x, \mathrm{SK}_f, \mathrm{PP}) = M \tag{6}$$

Definition 2.17. (Adaptive Security of KP-ABE.) Let $\mathcal{S} =$ (Setup, Encrypt, KeyGen, Decrypt) be a key-policy ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We define adaptive security to be the intractability assumption that consists of the following interactive "game":

1. **Setup Phase**   The challenger runs Setup(1^λ) and sends PP to the adversary.
2. **Key Query Phase I**   The adversary makes key queries for policies in $\mathcal{F}$ of her choice: that is, in each key query, the adversary sends a policy of her choice to the challenger, the challenger runs the KeyGen algorithm to produce a secret key and sends it to the adversary.
3. **Challenge Phase** The adversary declares two messages M_0, M_1 and a challenge attribute $x^* \in \mathcal{X}$. The challenger samples a uniformly random bit $b \in \{0,1\}$, computes $\mathrm{CT}_{x^*} = \mathrm{Encrypt}\,(x^*, M_b, \mathrm{PP})$ and sends the result to the adversary.
4. **Key Query Phase II**   Same as Key Query Phase I.

5. **Guess** The adversary sends a guess b' for the bit b.

The output of the game is defined as follows: If every queried policy f satisfies $f(x^*) = 0$, the output is the adversary's guess $\tilde{b} = b'$; otherwise, the output is a uniformly random bit $\tilde{b} \xleftarrow{\$} \{0,1\}$. The threshold of the intractability assumption is 1/2, so the advantage of the adversary is $\left|\Pr[\tilde{b} = b] - 1/2\right|$.

Remark 2.18. We observe that the adaptive security of an ABE scheme depends only on the Setup, KeyGen, and Encrypt procedures and not on the Decrypt procedure. Therefore, any two ABE schemes that differ only in their implementations of the decryption must either both be adaptively secure or both not.

Next, we formally define a condition on an ABE scheme that essentially requires the scheme to support a set of policies derived from a pairwise-independent hash family. Intuitively, a random sample of policies from such a set has a significant probability of accepting a uniformly random attribute. This property will be used in our proof to ensure that an attacker that requests secret keys for many policies in that set can use them to decrypt a challenge ciphertext with a large enough probability.

Definition 2.19. (Pairwise-Friendliness of ABE.) Let $\mathcal{S}$ be an ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ is pairwise friendly if for every $a, b \in \mathbb{N}$ such that $a > b$ and $a = O(\log|\mathcal{X}|)$, there exists a pairwise independent hash family $H = \{h : \mathcal{X} \to [a]\}$ so that the following holds: For every $h \in H$, the function $f_h : \mathcal{X} \to \{0,1\}$ defined by

$$f_h(x) = 1 \iff h(x) \leq b \tag{7}$$

is in $\mathcal{F}$.

Remark 2.20. For every $n \in \mathbb{N}$ and $m \leq n$ there exists a pairwise independent hash family $H = \{h : \{0,1\}^n \to \{0,1\}^m\}$ such that every $h \in H$ can be computed by an NC^1 circuit [IKOS08].

Remark 2.21. Following the previous remark, let $\mathcal{S}$ be an ABE scheme with attribute set $\mathcal{X}$ and policy class $\mathcal{F}$ such that $\mathcal{F}$ contains the class of functions with depth-d circuits for $d = O(\log|\mathcal{X}|)$, then $\mathcal{S}$ is pairwise friendly.

The following definitions describe our checkability criterion for an ABE scheme. Intuitively, the checkability condition requires a validation procedure for secret keys, which ensures that validated keys decrypt any ciphertext the same. In particular, it makes it impossible for a challenger to distinguish which key was used to decrypt a challenge ciphertext. More precisely, checkability correctness requires that honestly generated keys are indeed valid (according to the validation procedure), and checkability soundness requires that any two valid keys decrypt any ciphertext the same way.

Definition 2.22. (Checkability of ABE.) We say that an ABE scheme is "checkable" if it has an additional algorithm:

- **SKCheck**$(\mathrm{PP}, \mathrm{SK}, f) \to \{\mathrm{True}, \mathrm{False}\}$ The ciphertext checking algorithm takes in public parameters PP, a key SK and a policy $f \in \mathcal{F}$. It outputs either True or False.

Definition 2.23. (Checkability Correctness of KP-ABE.) Let $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt, SKCheck) be a KP-ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ satisfies the checkability-correctness property if the following holds: Let (PP, MSK) = $\mathrm{Setup}(1^\lambda)$, $f \in \mathcal{F}$, and $\mathrm{SK}_f = \mathrm{KeyGen}(\mathrm{MSK}, f, \mathrm{PP})$, then $\mathrm{SKCheck}(\mathrm{PP}, \mathrm{SK}_f, f) = \mathrm{True}$.

Definition 2.24. (Checkability Soundness of KP-ABE.) Let $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt, SKCheck) be a KP-ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ satisfies the checkability-soundness property if the following holds: Let PP, $\mathrm{SK}_1, \mathrm{SK}_2$ and $f_1, f_2 \in \mathcal{F}$. Let CT_x, i.e. a ciphertext claimed to be encrypted w.r.t. attribute $x \in \mathcal{X}$,[6]

[6]Recall our convention (Definition 2.14) that ciphertexts contain their attribute as a part of their description.

s.t. $f_1(x) = f_2(x) = 1$. If $\mathrm{SKCheck}(\mathrm{PP}, \mathrm{SK}_1, f_1) = \mathrm{SKCheck}(\mathrm{PP}, \mathrm{SK}_2, f_2) = \mathrm{True}$, then

$$\mathrm{Decrypt}(\mathrm{CT}_x, \mathrm{SK}_1, \mathrm{PP}) = \mathrm{Decrypt}(\mathrm{CT}_x, \mathrm{SK}_2, \mathrm{PP}) \tag{8}$$

Remark 2.25. To obtain our result, it suffices to assume a looser condition on a checkable ABE scheme: Instead of requiring that any two policy keys decrypt the ciphertext exactly the same, it suffices to require that for any two policy keys, the distributions of the decryption outputs are computationally indistinguishable.

3 The Main Theorem and Proof

Informally, our main theorem asserts the following: Any pairwise-friendly, checkable ABE scheme cannot be proven adaptively secure by constructing a black-box reduction that reduces an intractability assumption to breaking the security of the scheme, even if the reduction is rewinding. We prove that if such a reduction existed, it would be possible to construct an efficient algorithm that violates the intractability assumption by using only the reduction, without requiring a successful ABE adversary, thereby leading to a contradiction.

Theorem 3.1. Let λ a security parameter and $n = \mathrm{poly}(\lambda)$. Let $\mathcal{S}$ be a pairwise friendly and checkable ABE scheme with attribute set $\mathcal{X} = \{0,1\}^n$ and policy class $\mathcal{F}$. Let $\mathcal{C}$ be an $r(\cdot)$-round intractability assumption with threshold $t(\cdot)$, where r, t are polynomials. Suppose that for every polynomial $l(\cdot)$ there exists a black-box reduction $\mathcal{R}$ such that the following holds: If $\mathcal{R}$ is given oracle access to an attacker $\mathcal{A}$ that makes $l(\cdot)$ key queries and has a non-negligible advantage in the adaptive security game of $\mathcal{S}$, then $\mathcal{R}^{\mathcal{A}}$ has non-negligible advantage w.r.t. the assumption $\mathcal{C}$.

Let $l(\lambda) = \omega(n(\lambda) + r(\lambda))$ and a corresponding reduction $\mathcal{R}$. Denote $\mathcal{A}$ to be a hypothetical attacker that has a non-negligible advantage in the adaptive security game of $\mathcal{S}$. Then there exists a polynomial-time algorithm $\mathcal{B}$ and a negligible function $\mu(\cdot)$ such that

$$\left|\Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \left(1^\lambda\right) = 1\right] - \Pr\left[\langle \mathcal{B}^{\mathcal{R}}, \mathcal{C}\rangle \left(1^\lambda\right) = 1\right]\right| \leq \mu(\lambda) \tag{9}$$

In particular, $\mathcal{B}^{\mathcal{R}}$ is a polynomial-time algorithm that has a non-negligible advantage w.r.t. the assumption $\mathcal{C}$.

In the remainder of the section, we prove the theorem:

Proof. Let λ denote the security parameter and suppose there exist $\mathcal{S}$ and $\mathcal{C}$ as described in the theorem. Let $l(\lambda) = \omega(n(\lambda) + r(\lambda))$ and let $\mathcal{R}$ be the corresponding reduction as described in the theorem. By definition of $\mathcal{R}$, if there were an attacker $\mathcal{A}$ with non-negligible advantage in the security game of $\mathcal{S}$, and $\mathcal{R}$ would be given oracle access to $\mathcal{A}$, then $\mathcal{R}$ could use $\mathcal{A}$ during an interaction with $\mathcal{C}$ to gain non-negligible advantage w.r.t. $\mathcal{C}$. We will prove that the oracle access to $\mathcal{A}$ can be simulated in a way that preserves $\mathcal{R}$'s advantage w.r.t. $\mathcal{C}$, even if it does not have oracle access to an actual attacker $\mathcal{A}$.

More explicitly, we will construct a machine $\mathcal{B}$ that has oracle access to the reduction $\mathcal{R}$ and simulates the interaction between $\mathcal{C}$ and $\mathcal{R}^{\mathcal{A}}$ for a properly defined $\mathcal{A}$ so that $\mathcal{B}$ has non-negligible advantage w.r.t. $\mathcal{C}$. To simulate the interaction accurately, $\mathcal{B}$ emulates an oracle access to an attacker $\mathcal{A}$ for $\mathcal{R}$, making it appear as if $\mathcal{A}$ has a non-negligible advantage in the adaptive security game of $\mathcal{S}$. Furthermore, the expected running time of $\mathcal{B}$ is polynomial, and in particular, the emulation of the attacker is efficient.

First, we introduce an inefficient hypothetical attacker $\mathcal{A}$ with a non-negligible advantage in the adaptive security game of $\mathcal{S}$. $\mathcal{A}$ can be used by $\mathcal{R}$ so that $\mathcal{R}^{\mathcal{A}}$ has non-negligible advantage w.r.t. the assumption $\mathcal{C}$. Second, we describe the algorithm $\mathcal{B}$ that has oracle access to the reduction $\mathcal{R}$, but instead of giving $\mathcal{R}$ an oracle access to an actual attacker, it simulates an attacker for $\mathcal{R}$ that is indistinguishable from $\mathcal{A}$ by $\mathcal{R}$. As we will show, $\mathcal{B}$ simulates $\mathcal{A}$ by rewinding the reduction $\mathcal{R}$ and exploiting the fact that $\mathcal{R}$ runs in polynomial time and cannot make too many queries to $\mathcal{A}$. Finally, we analyze the running time and advantage of $\mathcal{B}^{\mathcal{R}}$ to prove that there exists a polynomial-time algorithm that breaks the assumption $\mathcal{C}$ with non-negligible advantage.

To summarize notations:

Parameter	Size	Meaning
n	$n(\lambda) = \text{poly}(\lambda)$	The dimension of $\mathcal{X} = \{0,1\}^n$, the attribute set of $\mathcal{S}$ on input 1^λ.
r	$r(\lambda) = \text{poly}(\lambda)$	The number of communication rounds in the intractability assumption $\mathcal{C}$ on input 1^λ.
t	$t(\lambda) = \text{poly}(\lambda)$	The threshold associated with the intractability assumption $\mathcal{C}$ on input 1^λ.
l	$l(\lambda) = \omega(n(\lambda) + r(\lambda))$	The number of key queries made by the attacker $\mathcal{A}$ on input 1^λ.
M	$M(\lambda) = \text{poly}(\lambda)$	The bound on the running time of $\mathcal{R}$ on input 1^λ.
m	$m(\lambda) > 8l(\lambda)$	A parameter of $\mathcal{B}$ to be defined later.
k	$k(\lambda) = \omega(l(\lambda) \log \lambda)$	A parameter of $\mathcal{B}$ to be defined later.

Recall that $\mathcal{S}$ is pairwise friendly, and let $H = \{h : \{0,1\}^n \to [m]\}$ be the pairwise independent hash family such that for every $h \in H$, the function f_h defined by $f_h(x) = 1 \iff h(x) \leq \frac{m}{8l}$ is in $\mathcal{F}$.

3.1 Hypothetical Attacker $\mathcal{A}$

The hypothetical attacker algorithm performs as follows:

Algorithm $\mathcal{A}(1^\lambda)$

1. Receive PP from the challenger.
2. Initialize $F = \emptyset$ the set of functions to be queried during the key query phase and their corresponding keys.
3. Make l key queries, for each one:
   (a) Sample a uniformly random $h \xleftarrow{\$} H$ and make a key query by sending the policy $f \stackrel{\text{def}}{=} f_h$ to the challenger.
   (b) Receive SK_f and update $F \leftarrow F \cup \{(f, \text{SK}_f)\}$.
   (c) Run SKCheck(PP, SK_f, f), if the output is False then abort.
4. Declare two messages $M_0 = 0$ and $M_1 = 1$. Sample a uniformly random challenge attribute $x^* \xleftarrow{\$} \{0,1\}^n$ and send it to the challenger. Receive a ciphertext CT_{x^*}.
5. Brute-force search for $h' \in H$ such that $f_{h'}(x^*) = 1$. If no such h' exists, then abort.
6. Iterate over all possible secret keys and check for every key SK if $\text{SKCheck}(\text{PP}, \text{SK}, f_{h'}) =$ True. If so, stop the iteration. If no such key was found, then abort.
7. Decrypt CT_{x^*} using SK and return the result.

Similarly to [Pas11], we formally equip $\mathcal{A}$ with access to a random oracle which is used to generate the random coins used by $\mathcal{A}$. This will allow us to consider rewindings of $\mathcal{R}$ more conveniently. We may think of $\mathcal{A}$ as a distribution over attackers, each defined by an instantiation of the random oracle.

Success Probability. We show that $\mathcal{A}$ breaks the adaptive security of $\mathcal{S}$:

Claim 3.2. $\mathcal{A}$ wins the adaptive security game of $\mathcal{S}$ with advantage $\geq \frac{7}{16}$.

Proof. We first observe that in the adaptive-security game of $\mathcal{S}$, all values (public parameters, ciphertexts, and keys) are honestly generated, so due to the checkability correctness of $\mathcal{S}$, all the checks of SKCheck in step (3c) pass. Furthermore, if step (5) does not abort, then by the definition of H and the checkability of $\mathcal{S}$ there necessarily exists a secret key SK such that $\mathrm{SKCheck}(\mathrm{PP}, \mathrm{SK}, f_{h'}) = \mathrm{True}$, so step (6) does not abort as well.

Next, we claim that there exists $h \in H$ such that $f_h(x^*) = 1$. By definition of pairwise independence of H, there is a non-zero probability to sample a random $h \xleftarrow{\$} H$ such that $f_h(x^*) = 1$. Therefore, step (5) never aborts.

So far, we conclude that if $\mathcal{A}$ samples a challenge x^* such that there is no $(f, \mathrm{SK}_f) \in F$ for which $f(x^*) = 1$, then $\mathcal{A}$ necessarily retrieves a secret key SK that decrypts CT_x and wins the game. We highlight the role of checkability soundness of $\mathcal{S}$: The retrieved key is validated using SKCheck, so the decryption of the ciphertext is the same as it would have been under any other validated secret key, therefore the reduction cannot distinguish which key was used to decrypt.

The probability to sample a challenge x^* such that there exists $(f, \mathrm{SK}_f) \in F$ for which $f(x^*) = 1$ is bounded by

$$\begin{aligned}\Pr_{x,F}\left[\exists (f, \mathrm{SK}_f) \in F \text{ s.t. } f(x) = 1\right] \leq l \cdot \Pr_{x,f}\left[f(x) = 1\right] = l \cdot \Pr_{h}\left[h(x) \leq \frac{m}{8l}\right] \\ \leq l \cdot \frac{1}{8l} = \frac{1}{8}\end{aligned} \tag{10}$$

When that happens, the winning probability is 1/2 by the definition of the security game. We conclude that the overall advantage is lower-bounded by $\frac{7}{16}$, as desired. □

3.2 Algorithm $\mathcal{B}$

Overview. Recall that we wish to construct an algorithm $\mathcal{B}$ that simulates oracle access to an attacker $\mathcal{A}$ for $\mathcal{R}$ without having access to an actual attacker. In such a simulation, $\mathcal{R}$ initiates an interaction with the attacker by first sending the public parameters; then they interact according to the adaptive security game of $\mathcal{S}$ until finally, the attacker sends a guess for which message was encrypted in the challenge ciphertext. Naively, $\mathcal{B}$ could simulate the attacker by rewinding $\mathcal{R}$ to a previous state of the interaction, extracting an additional secret key, and using it to decrypt the challenge ciphertext. We emphasize that the extracted secret key should be validated by the attacker using the SKCheck procedure to ensure canonical decryption, as guaranteed by the checkability soundness of the scheme. A major problem with this naive approach is that $\mathcal{R}$ can make "intertwined" queries to $\mathcal{A}$ for many different public parameters. Therefore, if $\mathcal{B}$ would rewind $\mathcal{R}$ whenever it would be required to decrypt a challenge ciphertext, we could get an exponential blow-up in running time[7]. We resolve this by having $\mathcal{B}$ rewind the reduction $\mathcal{R}$ only under certain conditions and have a more delicate rewinding process, as we will describe shortly.

Before diving into a formal description of the rewinding process, we provide a high-level intuition. Consider the tape of the Turing machine $\mathcal{B}^{\mathcal{R}}$, i.e., the execution of $\mathcal{B}$ given oracle access to $\mathcal{R}$. At a high level, $\mathcal{B}$ initiates an execution of $\mathcal{R}$, forwards all communication between $\mathcal{C}$ and $\mathcal{R}$, and whenever $\mathcal{R}$ tries to access the attacker oracle, $\mathcal{B}$ emulates the attacker for $\mathcal{R}$. Under certain conditions to be specified later, $\mathcal{B}$ "forks" the execution into two parallel

[7] A similar problem was presented in the context of concurrent zero-knowledge in [DNS04].

executions, a process which could be thought of as "duplicating" the machine tape, and rewinds $\mathcal{R}$ in the forked execution to a previous state. In other words, $\mathcal{B}$ makes a copy of the current state of execution and then rewinds the copied execution so that it would continue differently from the original one. We think of the relation between the original execution and the duplicated one as "parent" and "child", respectively. $\mathcal{B}$ then continues running the child execution until a certain event occurs, when it terminates it (and all its child executions if they exist), and finally continues the parent execution.

More precisely, taking inspiration from Pass's work [Pas11], we define the notion of a "slot", denoted by s, to be a time window within the execution of $\mathcal{R}$ that "opens" just before the simulated attacker sends a policy to the reduction, and "closes" right after the reduction sends back a corresponding secret key and the simulated attacker validates it using SKCheck. Whenever a slot closes, $\mathcal{B}$ decides whether to rewind $\mathcal{R}$ back to the opening of the slot, depending on three conditions that determine if the slot is "good":

1. Between the time the slot s opened and the time it closed, $\mathcal{R}$ did not send (and thus did not receive) any external message to (or from) $\mathcal{C}$.

2. Between the time the slot s opened and the time it closed, the number of other slots that opened is "small", where "small" will be defined below.

3. The received key is valid, i.e., the result of SKCheck was True.

Whenever such a slot s closes, $\mathcal{B}$ "duplicates" the execution, rewinds $\mathcal{R}$ in the duplicated execution back to the opening of the slot, and sends a different policy than the one sent in the original execution. $\mathcal{B}$ runs the duplicated execution until the slot either closes or stops being "good", and then terminates it (along with all its child executions if they exist). $\mathcal{B}$ repeats this process of duplicating the execution and rewinding the slot several times to extract several keys until finally, $\mathcal{B}$ returns to the original execution and continues running it. We highlight that the rewinding process could be recursive – recall that $\mathcal{R}$ might make intertwined queries, thus, there might be a slot that opens and closes within another slot.

Next, we describe this procedure formally.

The Algorithm. We will use the notion of a machine state, or simply a state, to formally describe the control flow of the algorithm. Intuitively, we could think of a state as a pointer to the machine tape of $\mathcal{B}$. W.l.o.g., we assume a state includes a record of all the messages sent and received by $\mathcal{C}$, $\mathcal{B}$ and $\mathcal{R}$, up to the point that the state points to.

We define a state v to be d-good with respect to a previous state u if: (1) $\mathcal{R}$ does not attempt to send messages to $\mathcal{C}$ during the time between u and v, and (2) the number of slots that open between u and v is at most $\frac{M}{n^d}$. We define a slot to be the time window whose opening is a state right before $\mathcal{B}$ makes sends key-query, and closing is a state right after the respective key that was received is checked using SKCheck. We observe that any slot can be specified by its opening state, and the opening defines a distribution over the possible closings of the slot (depending on the key that was queried and the respective response). An execution-instance of a slot is a pair (u, v) where u is the slot-opening and v is a specific slot-closing. We say that an execution-instance (u, v) of slot s is d-good if the closing of s, v, is d-good with respect to its opening, u, and the result of SKCheck at the end of the slot is True.

The algorithm we describe is $\mathcal{B}^{\mathcal{R}}$, that is, the algorithm $\mathcal{B}$ given oracle access to the reduction $\mathcal{R}$. The formal description uses the definition of the hypothetical attacker $\mathcal{A}$ and a recursive procedure SIM that simulates the attacker oracle for $\mathcal{R}$.

Recall that the interactive protocol between the reduction and the attacker oracle begins with the reduction sending public parameters to the attacker. Since $\mathcal{R}$ can make queries to the attacker that correspond to intertwined interaction transcripts, we associate each message with the public parameters that initiated the corresponding transcript. Similarly, we associate each slot with the public parameters that initiated the transcript that contains the policy message that "opened" the slot.

Algorithm $\mathcal{B}^{\mathcal{R}}(1^\lambda)$

1. Initialize a global set $\tilde{F} = \emptyset$.
2. Receive a message from $\mathcal{C}$.
3. Initiate an execution of the reduction oracle $\mathcal{R}$, and send the message received from $\mathcal{C}$ to $\mathcal{R}$.
4. Run $\mathrm{SIM}^{\mathcal{R}}(1^\lambda, 0, 0, 0)$.

Algorithm SIM$^{\mathcal{R}}(1^\lambda, d, u, v)$
On input the recursive depth d, a state u, and a state v:

1. Check for the following mutually exclusive conditions and perform accordingly:
   (a) If $d = 0$ and $\mathcal{R}$ attempts to send a message to $\mathcal{C}$, forward the message and feed $\mathcal{R}$ the response received from $\mathcal{C}$. Note that only at recursive depth $d = 0$ the reduction $\mathcal{R}$ can interact with $\mathcal{C}$.
   (b) If $d > 0$ and v is not a d-good state with respect to u, return $\perp$.
   (c) If $d > 0$, u is an opening of a slot s that closes at v, and (u, v) is a d-good execution-instance of s, then return $(\mathrm{PP}, f, \mathrm{SK}_f)$ where f and SK_f are the policy and secret key retrieved during the slot s and PP is the corresponding public parameters.
   (d) If v is the closing of a slot $\tilde{s}$ that opened at state $\tilde{u}$ which is strictly after u, and $(\tilde{u}, v)$ is a $(d+1)$-good execution-instance of $\tilde{s}$: Repeat the following $k(n)$ times:
      i. Let $r = \mathrm{SIM}(1^\lambda, d+1, \tilde{u}, \tilde{u})$.
      (We emphasize that this is where the rewinding occurs – another execution of SIM is forked and rewound to the previous state $\tilde{u}$.)
      ii. If $r \neq \perp$, store $r = (\mathrm{PP}, f, \mathrm{SK}_f)$ in $\tilde{F}$.
2. If none of the above conditions were satisfied, check if $\mathcal{R}$ is sending a message to the attacker:
   (a) If $\mathcal{R}$ is sending a challenge ciphertext, do nothing and continue.
   (b) If $\mathcal{R}$ is sending a secret key SK as a response to a key query for policy f with respect to public parameters PP:
      i. Run $\mathrm{SKCheck}(\mathrm{PP}, \mathrm{SK}, f)$.
      ii. If the output is False and $d = 0$ then abort. If the output is False and $d > 0$ then return $\perp$.
3. If none of the conditions of (1) and (2) were satisfied, check if according to the interactive protocol between $\mathcal{R}$ and the hypothetical attacker $\mathcal{A}$, $\mathcal{R}$ is expecting to receive a response message from the attacker:
   (a) If $\mathcal{R}$ is expecting to receive a decryption of a ciphertext associated with public parameters PP, then perform as follows: Let CT_{x^*} be the ciphertext and x^* the challenge under which the ciphertext was encrypted;

i. Search $\tilde{F}$ for a tuple that has the same PP and also satisfies $f(x^*) = 1$.

ii. If there exists such a tuple, use SK_f to decrypt CT_{x^*} and send the result.

iii. Otherwise, send a uniformly random guess.

(b) If $\mathcal{R}$ is expecting to receive a policy, then respond as the attacker oracle would; that is, sample a uniformly random $h \xleftarrow{\$} H$ and send f_h.

(c) If $\mathcal{R}$ is expecting to receive a M_0, M_1 and a challenge attribute, then respond as the hypothetical attacker oracle would; that is,

i. Declare two messages $M_0 = 0$ and $M_1 = 1$.

ii. Ssample a uniformly random $x^* \xleftarrow{\$} \{0,1\}^n$ and send it to $\mathcal{R}$.

4. Update v to be the current state (that includes all messages up to the current point) and return $\mathrm{SIM}^{\mathcal{R}}\left(1^\lambda, d, u, v\right)$.

3.3 Running Time

Consider a variant of $\mathcal{B}$, $\tilde{\mathcal{B}}$, such that whenever $\tilde{\mathcal{B}}$ is required to decrypt a challenge ciphertext without having retrieved a suitable secret key, $\tilde{\mathcal{B}}$ magically gets a key that decrypts the ciphertext (we can think of this as brute-force searching for a key in the same manner as the hypothetical attacker, but without counting the brute-force search into the runtime). We note that this change does not change the runtime of the algorithm. We observe that $\tilde{\mathcal{B}}$ always responds the same as the hypothetical attacker $\mathcal{A}$.

Lemma 3.3. There exists some polynomial $p(\cdot)$ such that the running time of $\tilde{\mathcal{B}}^{\mathcal{R}}\left(1^\lambda\right)$ is bounded by $p(\lambda)$.

Proof. We use a recursion tree to describe how $\tilde{\mathcal{B}}$ executes and rewinds $\mathcal{R}$: The root of the tree is the initial (and single) execution of $\mathcal{R}$ at level $d = 0$. Whenever $\tilde{\mathcal{B}}$ forks a child execution at recursive level d, that is, what we previously described as "duplicating the tape", it is translated into a new node at level $d+1$ which is a child of the node from which it was forked from at level d. As previously described, this occurs whenever $\tilde{\mathcal{B}}$ encounters a $(d+1)$-good slot instance (i.e., a $(d+1)$-good execution-instance of a slot) during an execution at recursive level d, which $\tilde{\mathcal{B}}$ then rewinds in the forked execution.

First, by definition of the algorithm $\tilde{\mathcal{B}}$, the depth of the recursion tree is bounded by a constant $c = \log_\lambda M \cdot \theta(1)$. Second, at each execution of $\mathcal{R}$ at recursive level d, $\mathcal{R}$ opens at most M slots, so there are at most M points from which $\tilde{\mathcal{B}}$ may fork child executions. Third, also by definition of $\mathcal{B}$, each slot is forked and rewound k times. Combining these observations, we get that each execution node in the recursion tree has at most M slots that have k children each, so overall a maximal number of Mk child executions. Therefore, the overall size of the recursion tree is bounded by $(Mk)^{c+1}$, thus the runtime of $\tilde{\mathcal{B}}^{\mathcal{R}}$ is bounded by a polynomial of λ as well. □

To conclude a bound on the expected running time of $\mathcal{B}$ from the bound on the running time of $\tilde{\mathcal{B}}$, we first make the following assumption: Suppose that the probability of $\mathcal{B}$ being required to decrypt a ciphertext without having retrieved a suitable key during an execution of $\mathcal{B}^{\mathcal{R}}$ is bounded by a negligible function of λ. Given this assumption, the transcript of the interaction between $\mathcal{B}^{\mathcal{R}}$ with $\mathcal{C}$ is indistinguishable from the transcript of the interaction between $\tilde{\mathcal{B}}^{\mathcal{R}}$ with $\mathcal{C}$. Therefore, under this assumption, the expected running time of $\mathcal{B}$ is bounded by a polynomial as well. The assumption is proven in the next section, in which we analyze the success probability of $\mathcal{B}$ breaking the intractability assumption $\mathcal{C}$.

3.4 Success Probability.

In order to analyze the success probability of $\mathcal{B}^{\mathcal{R}}$, we compare the transcript of the interaction between $\mathcal{C}$ and $\mathcal{B}^{\mathcal{R}}$ with the transcript of the interaction between $\mathcal{C}$ and $\mathcal{R}^{\mathcal{A}}$. We show that the distributions of those transcripts are indistinguishable, therefore, the probability that $\mathcal{C}$ outputs 1 is the same in both scenarios except for some negligible probability. In other words, we show that there exists some negligible function $\mu(\cdot)$ such that

$$\Pr\left[\langle \mathcal{B}^{\mathcal{R}}, \mathcal{C}\rangle \left(1^{\lambda}\right) = 1\right] \geq \Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \left(1^{\lambda}\right) = 1\right] - \mu(\lambda) \tag{11}$$

By the definition of $\mathcal{B}$, it forwards all messages from $\mathcal{C}$ to the (single) execution of $\mathcal{R}$ at recursive level $d = 0$ and vice versa. For simplicity, denote the execution of $\mathcal{R}$ at level $d = 0$ by $\mathcal{R}_0$. If $\mathcal{B}$ would perfectly simulate the hypothetical attacker $\mathcal{A}$ for $\mathcal{R}$, that is, respond to all queries made by $\mathcal{R}$ to the attacker oracle exactly the same as $\mathcal{A}$ would, then $\mathcal{R}_0$ would behave exactly the same as $\mathcal{R}^{\mathcal{A}}$ (all other recursive calls at level $d > 0$ would be irrelevant to $\mathcal{R}_0$), leading to the transcripts in both scenarios having exactly the same distribution. Although this is not necessarily the case, we show that the probability that $\mathcal{B}$ responds differently from $\mathcal{A}$ is negligible, so even though the transcripts are not identically distributed, they are indeed indistinguishable.

By the definitions of the hypothetical attacker $\mathcal{A}$ and the algorithm $\mathcal{B}$, they respond exactly the same to all queries that $\mathcal{R}$ makes to the attacker oracle, with the exception of when $\mathcal{R}$ requires the attacker oracle to provide a guess for which message was encrypted in the challenge ciphertext. When that happens, then the hypothetical attacker should be able to decrypt by brute-force searching for a decrypting secret key, whereas the simulated attacker might fail to obtain such a key and respond differently than $\mathcal{A}$. We argue that the probability that $\mathcal{B}$ fails to decrypt a ciphertext is negligible.

Denote:

- E_1 the event that $\mathcal{B}$ is required to decrypt a ciphertext CT_x associated with some public parameters PP without having previously rewound at least n slots associated with the same PP.

- E_2 the event that $\mathcal{B}$ is required to decrypt a ciphertext CT_x associated with some public parameters PP, at least n slots associated with the same PP were successfully rewound, but every $(\mathrm{PP}, f, \mathrm{SK}_f) \in \tilde{F}$ (with the same PP) satisfies $f(x) = 0$.

If neither E_1 nor E_2 occur, then $\mathcal{B}$ necessarily obtains a decrypting key and successfully decrypts the challenge ciphertext.

Claim 3.4. The probability that E_1 happens in an execution between $\mathcal{B}^{\mathcal{R}}$ and $\mathcal{C}$ is 0.

Proof. The proof of this claim follows similar guidelines as presented in [Pas11]. We first observe that every time an instance of a game between $\mathcal{R}$ and the simulated attacker reaches the challenge phase, and $\mathcal{R}$ sends a challenge ciphertext associated with public parameters PP, then it must have sent $l(\lambda) = \omega(n+r) = \omega(2n+r+1)$ secret keys[8] associated with the same PP (i.e., as part of the same game). Moreover, those keys must have been valid ones, otherwise, the simulated attacker would have stopped responding and never reached the guessing phase.

Consider an occurrence of $\mathcal{B}$ being required to send a guess as part of a game associated with public parameters PP, and the l slots associated with the same PP. These slots may be distributed over several nodes in the recursion tree. Since the recursive depth of the simulation is bounded by a constant c, there must be some recursion level d such that the number of these slots at level d is at least $\frac{l}{c}$. For sufficiently large λ, $\frac{l}{c} \geq 2n + r + 1$. Since r bounds the total number of external messages, we conclude that in at least $2n + 1$ of the slots, there is no

[8]It will be more convenient to think about $(2n + r + 1)$ as a fundamental quantity as we see below.

communication between $\mathcal{R}$ and $\mathcal{C}$. By the design of the rewinding process, the number of slot openings during any slot s at level d is bounded by $\frac{M}{n^d}$, so there must be at least n of the l/c slots that have no more than $\frac{M}{n^{d+1}}$ inner slot openings. We conclude that at least n of the slots associated with PP are $(d+1)$-good, and thus can be rewound, as desired. $\square$

Claim 3.5. There exists some negligible function $\mu_2(\cdot)$ such that the probability that E_2 happens in an execution between $\mathcal{B}^{\mathcal{R}}$ and $\mathcal{C}$ on input 1^λ is bounded by $\mu_2(\lambda)$.

Proof. Consider a query that $\mathcal{R}$ makes to the attacker and the respective security game, and assume that $\mathcal{B}$ is required to decrypt a challenge ciphertext. By Claim 3.4, there are at least n relevant slots that were rewound. Let $s_1, \ldots, s_n$ be those slots. For every $i \in [n]$ we let $W_i \subseteq H$ be the set of hash functions $h \in H$ such that querying a secret key for a policy f_h in slot s_i results in the reduction responding with a valid key SK_{f_h} with probability at least $(1-\alpha)$, where α is a parameter that we will choose shortly. Note that the sets $W_1, \ldots, W_n$ are random variables that depend on the random coins used by $\mathcal{R}$ and $\mathcal{B}$.

Denote by E_3 the event that $W_1, \ldots, W_n$ are all smaller than $|H|/4$. Since $\mathcal{B}$ samples the policies in the key queries uniformly at random, and the game reaches the challenge phase only if the reduction responds with valid keys, the probability that $\mathcal{B}$ is required to decrypt and E_3 occurs is bounded by

$$\begin{aligned} \Pr[(\mathcal{B} \text{ required to decrypt}) \wedge E_3] &\leq \prod_{i=1}^{n} \left[\frac{|W_i|}{|H|} \cdot 1 + \left(1 - \frac{|W_i|}{|H|}\right)(1-\alpha)\right] \\ &\leq \prod_{i=1}^{n} \left[(1-\alpha) + \frac{|W_i|}{|H|} \cdot \alpha\right] < \prod_{i=1}^{n} \left(1 - \frac{3}{4}\alpha\right) \end{aligned} \tag{12}$$

Choosing $\alpha = 1/4$, we get

$$\Pr[(\mathcal{B} \text{ required to decrypt}) \wedge E_3] \leq \left(\frac{13}{16}\right)^n \tag{13}$$

Thus, this probability is bounded by a negligible function.

From now on, we consider the case in which E_3 does not occur, that is, there is a slot s_i during the query phase that was successfully rewound and satisfies $|W_i| \geq |H|/4$. To analyze the probability that $\mathcal{B}$ fails to retrieve a key that decrypts the challenge ciphertext, we use the following additional notations:

- Let $T = \{1, \ldots, \lfloor \frac{m}{4l} \rfloor\}$, which is the set of values such that $h(x) \in T \iff f_h(x) = 1$.
- Let $W = W_i$, that is, the set of hash functions $h \in H$ such that querying f_h in slot s_i results in the reduction responding with a valid key with probability at least $1 - \alpha$ (where $\alpha = 1/4$).
- For all δ, let $S_\delta = \left\{x \in \{0,1\}^n \;\middle|\; \Pr_{h \xleftarrow{\$} W}[h(x) \in T] < \delta\right\}$, i.e., the set of all challenges such that if we sample a uniformly random $h \xleftarrow{\$} W$, the probability that $h(x) \in T$ is $< \delta$.
- For all δ and all $Y \subseteq H$ let X_Y^δ be the random variable $|h^{-1}(T) \cap S_\delta|$ when sampling a uniformly random $h \xleftarrow{\$} Y$.

Before diving further into the technical details, we provide a brief intuition. By definition of S_δ, this is intuitively a set of "bad challenges" – challenges that have a small probability $(< \delta)$ to be covered by a random hash function in W. Note that the smaller δ is, the smaller the set S_δ is. Since the attacker samples random hash functions in H, which is a pairwise independent family, the fraction of S_δ that is covered by a uniformly random $h \in H$ is close to its expectation with high probability. Therefore, for small δ, we expect to find only a small fraction of the functions in H whose intersection with S_δ is very small. These are intuitively "bad functions" because their contribution to the coverage of S_δ, the set of "bad challenges", is little.

To compute the probability of covering a random challenge attribute x^*, we will compute lower and upper bounds on the expected coverage of S_δ by a random function in W, and extract a trade-off between the size of S_δ and $1/\delta$. The lower bound computation will assume a worst-case scenario in which all of the "bad functions" are in W and exploit the fact that W is a large enough fraction of H to not be affected "too much" by the "bad functions" (when δ is chosen to be small enough).

As we will see, we can choose δ such that $\delta = 1/\text{poly}(\lambda)$ and the size of S_δ is negligible. From this, we will conclude that a random challenge attribute is covered by the keys retrieved from the k rewindings of s_i with all but negligible probability.

We continue with a formal analysis:

Lemma 3.6. For every δ we have

$$\delta > \frac{|T|}{2m}\left(1 - \frac{4|H|m}{|W||T||S_\delta|}\right) \tag{14}$$

Proof. By the definition of X_Y^δ, for every δ and every $Y \subseteq H$

$$\begin{aligned}\mathbb{E}\left[X_Y^\delta\right] &= \mathop{\mathbb{E}}_{h \xleftarrow{\$} Y}\left[|h^{-1}(T) \cap S_\delta|\right] = \mathop{\mathbb{E}}_{h \xleftarrow{\$} Y}\left[\sum_{x \in \{0,1\}^n} 1_{x \in S_\delta} \cdot 1_{h(x) \in T}\right] \\ &= \mathop{\mathbb{E}}_{h \xleftarrow{\$} Y}\left[\sum_{x \in S_\delta} 1_{h(x) \in T}\right]\end{aligned} \tag{15}$$

By the definition of X_H^δ,

$$\mathbb{E}\left[X_H^\delta\right] = \mathop{\mathbb{E}}_{h \xleftarrow{\$} H}\left[\sum_{x \in S_\delta} 1_{h(x) \in T}\right] = \sum_{x \in S_\delta} \Pr_{h \xleftarrow{\$} H}\left[h(x) \in T\right] = \frac{|T| \cdot |S_\delta|}{m} \tag{16}$$

$$\mathbb{E}\left[\left(X_H^\delta\right)^2\right] = \mathop{\mathbb{E}}_{h \xleftarrow{\$} H}\left[\left(\sum_{x \in S_\delta} 1_{h(x) \in T}\right)^2\right] \tag{17}$$

$$= \mathop{\mathbb{E}}_{h \xleftarrow{\$} H}\left[\sum_{x \in S_\delta} 1_{h(x) \in T} + \sum_{\substack{x,y \in S_\delta \\ x \neq y}} 1_{h(x) \in T} 1_{h(y) \in T}\right] \tag{18}$$

$$= \frac{|T||S_\delta|}{m} + \frac{|T|^2 \cdot |S_\delta|(|S_\delta| - 1)}{m^2} \tag{19}$$

$$\text{Var}\left[X_H^\delta\right] = \mathbb{E}\left[\left(X_H^\delta\right)^2\right] - \mathbb{E}\left[X_H^\delta\right]^2 = \frac{|T||S_\delta|}{m}\left(1 - \frac{|T|}{m}\right) \tag{20}$$

Applying Chebyshev's inequality, we get

$$\Pr\left[\left|X_H^\delta - \frac{|T||S_\delta|}{m}\right| \geq \frac{|T||S_\delta|}{2m}\right] \leq \frac{|T||S_\delta|}{m} \cdot \frac{4m^2}{|T|^2|S_\delta|^2} = \frac{4m}{|T||S_\delta|} \tag{21}$$

$$\Rightarrow \Pr\left[X_H^\delta \leq \frac{|T||S_\delta|}{2m}\right] \leq \frac{4m}{|T||S_\delta|} \tag{22}$$

In other words, the fraction of S_δ that is covered by a uniformly random $h \xleftarrow{\$} H$, which is $\frac{X_H^\delta}{|S_\delta|}$, is close its expectation $\frac{|T|}{m}$.

Let V_δ be the set of hash functions $h \in H$ such that $|h^{-1}(T) \cap S_\delta| < \frac{|T||S_\delta|}{2m}$. Intuitively, this is the set of "bad functions" because their contribution to the coverage of S_δ is much smaller than the expected coverage of a uniformly random $h \in H$. Note that by Eq. (22)

$$\begin{aligned} |V_\delta| &\leq |H| \cdot \Pr_{h \xleftarrow{\$} H}\left[|h^{-1}(T) \cap S_\delta| < \frac{|T||S_\delta|}{2m}\right] \\ &= |H| \cdot \Pr\left[X_H^\delta \leq \frac{|T||S_\delta|}{2m}\right] \leq \frac{4|H|m}{|T||S_\delta|} \end{aligned} \tag{23}$$

By the law of total expectation,

$$\begin{aligned} \mathbb{E}\left[X_W^\delta\right] \geq \mathbb{E}\left[X_{W\setminus V_\delta}^\delta\right] \cdot \Pr_{h \xleftarrow{\$} W}[h \notin V_\delta] \geq \frac{|T||S_\delta|}{2m} \cdot \left(1 - \frac{|V_\delta|}{|W|}\right) \\ \geq \frac{|T||S_\delta|}{2m} \cdot \left(1 - \frac{4|H|m}{|W||T||S_\delta|}\right) \end{aligned} \tag{24}$$

By definition of W and S_δ, we get the following upper bound

$$\mathbb{E}\left[X_W^\delta\right] = \mathbb{E}_{h \xleftarrow{\$} W}\left[\sum_{x \in S_\delta} 1_{h(x) \in T}\right] < \delta \cdot |S_\delta| \tag{25}$$

Combining equations (25) and (24), we get

$$\delta > \frac{|T|}{2m}\left(1 - \frac{4|H|m}{|W||T||S_\delta|}\right) \tag{26}$$

□

Next, we use lemma 3.6 and substitute the bounds $|W| \geq \frac{|H|}{4}$ and $\frac{|T|}{m} \geq \frac{1}{8l}$. We get that for every δ

$$\delta > \frac{1}{16l}\left(1 - \frac{2^7 \cdot l}{|S_\delta|}\right) \tag{27}$$

Setting $\delta' = \frac{1}{32l}$ and substituting into (27),

$$\frac{1}{32l} > \frac{1}{16l}\left(1 - \frac{2^7 \cdot l}{|S_{\delta'}|}\right) \Rightarrow |S_{\delta'}| < 2^8 \cdot l \tag{28}$$

For sufficiently large λ it holds that $l < 2^{n/2-8}$, thus $|S_{\delta'}| < 2^{n/2}$.

Next, we upper bound the probability of not covering a uniformly random $x^* \xleftarrow{\$} \{0,1\}^n$ by $\tilde{F}$, that is, the probability that $h(x^*) \notin T$ for every $f_h \in \tilde{F}$.

Let E_4 be the event that $x^* \in S_{\delta'}$, then we immediately get that

$$\Pr[E_4] \leq \frac{|S_{\delta'}|}{2^n} \leq 2^{-n/2} \tag{29}$$

In other words, E_4 occurs with negligible probability. Therefore, consider the case E_4 does not occur, meaning $x^* \notin S_{\delta'}$.

By definition of $S_{\delta'}$, since $x^* \notin S_{\delta'}$, at least δ' of the hash functions in W cover x^*. The probability that a key query results in a valid key that covers x^* is lower-bounded by

$$\begin{aligned}
&\Pr[\text{A key query results in a valid key that covers } x^*] \\
&\geq \Pr[\text{Sampling } h \in W \wedge \text{Receiving a valid key} \wedge \text{The sampled } h \in W \text{ covers } x^*] \\
&\geq \frac{|W|}{|H|} \cdot (1-\alpha) \cdot \delta' \geq \frac{3}{16}\delta'
\end{aligned} \tag{30}$$

Let E_5 be the event that none of the k rewound key queries (i.e., the key queries of the executions forked from rewinding slot s_i) result in a valid key that covers the uniformly random x^*, then

$$\begin{aligned}
\Pr[E_5] &= (1 - \Pr[\text{A key query results in a valid key that covers } x^*])^k \\
&\leq \left(1 - \frac{3}{16}\delta'\right)^k = \left(1 - \frac{3}{2^9 l}\right)^k \leq e^{-\frac{3k}{2^9 l}}
\end{aligned} \tag{31}$$

To summarize the computation, for every query that $\mathcal{R}$ makes to the attacker oracle, the probability of the game reaching the challenge phase and $\mathcal{B}$ failing to decrypt is bounded by

$$\begin{aligned}
\Pr[\mathcal{B} \text{ fails to decrypt}] &\leq \Pr[(\mathcal{B} \text{ required to decrypt}) \wedge E_3] + \Pr[E_4] + \Pr[E_5] \\
&\leq \left(\frac{13}{16}\right)^n + 2^{-n/2} + e^{-\frac{3k}{2^9 l}}
\end{aligned} \tag{32}$$

which is a negligible function of λ, as desired. $\square$

By the two previous claims, the attacker simulated by $\mathcal{B}$ is indistinguishable by $\mathcal{R}$ from the hypothetical attacker $\mathcal{A}$, therefore the transcript of the interaction between $\mathcal{C}$ and $\mathcal{R}_0$ is indistinguishable from the transcript of the interaction between $\mathcal{C}$ and $\mathcal{R}^{\mathcal{A}}$, and so

$$\Pr\left[\langle \mathcal{B}^{\mathcal{R}}, \mathcal{C}\rangle \left(1^\lambda\right) = 1\right] = \Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \left(1^\lambda\right) = 1\right] - \text{negl}(\lambda) \tag{33}$$

Combining with the runtime analysis, we conclude that the expected running time of $\mathcal{B}$ is bounded by a polynomial function of λ, thus by Markov's inequality, we can truncate $\mathcal{B}$ to run in strictly polynomial time while preserving its non-negligible advantage w.r.t. the assumption $\mathcal{C}$. Finally, we conclude that there exists a polynomial-time machine such that, if given oracle access to $\mathcal{R}$, has a non-negligible advantage w.r.t. the assumption $\mathcal{C}$. This completes the proof. $\square$

4 The Case of Lattice-Based ABE

As an example of applying our framework, we consider the celebrated [BGG+14] KP-ABE candidate and show that a its delegatable version conforms with the conditions of our main theorem. The [BGG+14] scheme has been proven selectively secure based on the hardness of Learning with Errors (LWE), and while we are not aware of it being conjectured adaptively secure, we do not know of concrete adaptive attacks. We consider a variant of the scheme where function secret keys consist of lattice trapdoors. This version can be adapted to our framework fairly straightforwardly.

4.1 Lattice Cryptography Background

We start by presenting a few necessary definitions of lattice cryptography on the subjects of LWE and lattice trapdoors, which are required in order to describe the ABE scheme formally.

Definition 4.1 (Decisional $\mathrm{LWE}_{n,m,q,\chi}$)**.** Let λ be a security parameter, $n = n(\lambda)$, $m = m(\lambda)$ and $q = q(\lambda)$ be integers, and $\chi = \chi(\lambda)$ be a noise distribution over $\mathbb{Z}$. The (n, m, q, χ)-LWE decision problem is to distinguish between the following two distributions: Letting $A \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, $s \xleftarrow{\$} \mathbb{Z}_q^n$, $e \leftarrow \chi^m$, $u \xleftarrow{\$} \mathbb{Z}_q^m$, the first distribution is $(A, A^T s + e)$ and the second is (A, u).

Definition 4.2 (Gadget Matrix)**.** We define the "gadget matrix" by $G = g \otimes I_n \in \mathbb{Z}_q^{n\times n\lceil\log q\rceil}$ where $g = (1, 2, 4, \dots, 2^{\lceil\log q\rceil-1}) \in \mathbb{Z}_q^{\lceil\log q\rceil}$. We define the inverse of the gadget matrix function $G^{-1} : \mathbb{Z}_q^{n\times m} \to \{0,1\}^{n\lceil\log q\rceil\times n}$ which expends each entry $a \in \mathbb{Z}_q$ of the input matrix into a column of size $\lceil\log q\rceil$ which is a binary representation of a, so for any matrix $A \in \mathbb{Z}_q^{n\times m}$ it holds that $G \cdot G^{-1}(A) = A$.

Definition 4.3 (Matrix Norms)**.** Let $T \in \mathbb{Z}^{n\times m}$ be a matrix and $\tilde{T}$ the result of applying Gram-Schmidt orthogonalization to the columns of T. We define the GS-norm $\|T\|_{\mathrm{GS}}$ as the l_2 length of the longest column of $\tilde{T}$. We let $\|T\|_2$ be the operator norm of T defined by $\|T\|_2 = \sup_{\|x\|=1} \|Tx\|$.

The following are properties of lattice trapdoors, see [BGG+14] for references.

Lemma 4.4. Let $m, n, q > 0$ be integers with q prime.

- There is an efficient randomized algorithm $\mathrm{TrapGen}(1^n, 1^m, q)$ that when $m = \Theta(n \log q)$, outputs a full-rank matrix $A \in \mathbb{Z}_q^{n\times m}$ along with a basis $T_A \in \mathbb{Z}^{m\times m}$ for
$$\Lambda_q^{\perp}(A) = \{z \in \mathbb{Z}^m \mid A \cdot z = 0 \mod q\}$$
such that A is statistically indistinguishable from a uniformly-sampled matrix and $\|T_A\|_{\mathrm{GS}} = O(\sqrt{n \log q})$, with all but negligible probability.
- There is an efficient algorithm $\mathrm{TrapExtend}(A, B, T_A)$ that given a full-rank matrix $A \in \mathbb{Z}_q^{n\times m}$, a basis T_A of $\Lambda_q^{\perp}(A)$, and $B \in \mathbb{Z}_q^{n\times m}$, outputs a basis $T_{[A|B]}$ of $\Lambda_q^{\perp}([A \mid B])$ such that $\|T_{[A|B]}\|_{\mathrm{GS}} = \|T_A\|_{\mathrm{GS}}$.
- There is a randomized algorithm $\mathrm{SampleD}(A, D, T_A, \sigma)$ that given a full-rank matrix $A \in \mathbb{Z}_q^{n\times m}$, a basis T_A of $\Lambda_q^{\perp}(A)$, a matrix $D \in \mathbb{Z}_q^{n\times k}$, and $\sigma = \|T_A\|_{\mathrm{GS}} \cdot \omega(\sqrt{\log m})$, outputs a random matrix $R \in \mathbb{Z}^{m\times k}$ such that $AR = D$, and $\|R^T\|_2 < m\sigma$ with all but negligible probability.

Lemma 4.5. Let $A \in \mathbb{Z}_q^{n\times m}$ be a full rank matrix with a basis $T_A \in \mathbb{Z}^{m\times m}$ for $\Lambda_q^{\perp}(A)$, and let $B \in \mathbb{Z}_q^{n\times k}$ and $C \in \mathbb{Z}_q^{k\times m}$ such that $A = BC$. Let $D \in \mathbb{Z}_q^{m\times k}$ such that $AD = B$ (which necessarily exists since A has a trapdoor T_A). Then the matrix $T_B = [CT_A \mid I - CD]$ is a full-rank set of vectors in $\Lambda_q^{\perp}(B)$ such that $\|T_B\|_{\mathrm{GS}} = \|CT_A\|_{\mathrm{GS}}$.

Proof. We can immediately verify that $BT_B = 0$, thus $T_B \in \Lambda_q^{\perp}(B)$. T_B is also full-rank since
$$T_B \begin{bmatrix} T_A^{-1}D \\ I \end{bmatrix} = I \tag{34}$$
For the same reason, the GS-norm of $T_B\|_{\mathrm{GS}}$ is the same as $[CT_A \mid I]$, which is $\|CT_A\|_{\mathrm{GS}}$, as desired. □

Key-Homomorphic Evaluation. Let f be a boolean circuit of depth d computing a function $\{0,1\}^k \to \{0,1\}$, and assume that f contains only NAND gates. We "translate" the operation of f into a computation on matrices: We associate with every input wire of f a matrix A_i, and for every other wire we assign a matrix recursively as follows: Let A_α, A_β be the matrices of the input wires, then the output wire is associated with the matrix $A_\gamma = A_\alpha \cdot G^{-1}(A_\beta) - G$. Note that for every input values $x_\alpha, x_\beta \in \{0,1\}$,

$$\begin{aligned}[A_\alpha + x_\alpha G \mid A_\beta + x_\beta G] \cdot \begin{bmatrix} G^{-1}(A_\beta) \\ -x_\alpha I \end{bmatrix} &= A_\gamma + (1 - x_\alpha x_\beta)G \\ &= A_\gamma + \mathrm{NAND}(x_\alpha, x_\beta)G \end{aligned} \tag{35}$$

Denote by A_f the matrix of the output wire of f. We define $\mathrm{Eval}(f, (A_1, \ldots, A_k))$ to be the procedure that takes as inputs f and $\vec{A} = (A_1, \ldots, A_k)$ and outputs A_f. Note that for input wires $x_1, \ldots, x_k$, the homomorphic evaluation satisfies

$$[A_1 + x_1 G \mid \cdots \mid A_k + x_k G] \cdot H_{f,x,\vec{A}} = A_f + f(x_1, \ldots, x_k)G \tag{36}$$

for some short matrix $H_{f,x,\vec{A}} \in \mathbb{Z}^{mk \times m}$ that has norm $O(n \log q)^{O(d)}$.

4.2 The [BGG+14] Scheme

Next, we describe the properties of [BGG+14] scheme and show their sufficiency for applying our theorem. For simplicity, we assume that the policies of the scheme accept attributes if and only if $f(x) = 0$ (instead of $f(x) = 1$).

Let λ denote the security parameter, the parameters of the scheme are an integer $n = n(\lambda)$, a prime $q = q(\lambda)$, an integer $m = \Theta(n \log q)$, a noise distribution $\chi = \chi(\lambda)$ over $\mathbb{Z}$, and $d = d(\lambda)$. The noise distribution is chosen to be $\chi_{\max}$-bounded, that is, its support is in $[-\chi_{\max}, \chi_{\max}]$. The attribute set is $\mathcal{X} = \{0,1\}^k$ for k which is given as input, and the class of policies $\mathcal{F}$ is the class of functions with depth-d circuits.

The public parameters of the scheme are matrices $A_0, A_1, \ldots, A_k, D \in \mathbb{Z}_q^{n \times m}$. Every attribute x is associated with a public key $A_x \in \mathbb{Z}_q^{n \times m}$ computed by

$$A_x \stackrel{\text{def}}{=} [A_0 \mid A_1 + x_1 G \mid \cdots \mid A_k + x_k G] \in \mathbb{Z}_q^{n \times m(k+1)} \tag{37}$$

Every predicate f is also associated with a public key $A_f \in \mathbb{Z}_q^{n \times m}$, computed by $\mathrm{Eval}(f, (A_1, \ldots, A_k))$.

At a high level, the encryption procedure of the scheme is a variant of dual Regev encryption [Reg05], so a ciphertext encrypted under a public key $A \in \mathbb{Z}_q^{n \times l}$ is essentially a noisy vector close to the lattice spanned by A, and has the form $c^T = s^T A + e^T$ where $s \xleftarrow{\$} \mathbb{Z}_q^n$ is a uniformly random vector and $e \in \mathbb{Z}^l$ is a noise vector sampled from a distribution over short vectors. A message is encoded into the ciphertext by adding an "offset" that depends on the message. A secret key SK for A is a lattice trapdoor T_A, i.e., a low-norm basis for the dual lattice $\Lambda_q^\perp(A)$. The trapdoor can be used to decrypt a ciphertext so long as the norm of its noise vector e is small enough.

Formally, the encryption of a message $M \in \{0,1\}^m$ under public key A_x is

$$\begin{aligned} \mathrm{CT}_x^T = & \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\mathrm{out}}^T\right] \\ = & \; s^T \left[A_0 \mid x_1 G + A_1 \mid \cdots \mid x_k G + A_k \mid D\right] \\ & + \left[e_0^T \mid e_1^T \mid \cdots \mid e_k^T \mid e_{\mathrm{out}}^T + \lceil q/2 \rceil M^T\right] \end{aligned} \tag{38}$$

where $s \xleftarrow{\$} \mathbb{Z}^n$ and $e_0, \ldots, e_k, e_{\mathrm{out}} \leftarrow \chi^m$.

To decrypt a ciphertext CT_x using a key SK_f for which $f(x) = 0$, one first homomorphically evaluates the ciphertext by applying a publicly known low-norm matrix $H_{f,x,\vec{A}} \in \mathbb{Z}^{mk\times m}$ (described in the key-homomorphic evaluation) that satisfies

$$[A_1 + x_1G \mid \cdots \mid A_k + x_kG]H_{f,x,\vec{A}} = A_f + f(x)G \in \mathbb{Z}_q^{n\times m} \tag{39}$$

The result of evaluating a ciphertext with respect to policy f is an encryption of the original message under the public matrix $[A_0 \mid A_f + f(x)G]$ as follows:

$$\begin{aligned} c_f^T &= \left[c_1^T \mid \cdots \mid c_k^T\right] H_{f,x,\vec{A}} \\ &= s^T \left[x_1G + A_1 \mid \cdots \mid x_kG + A_k\right] H_{f,x,\vec{A}} + \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\vec{A}} \\ &= s^T A_f + \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\vec{A}} \end{aligned} \tag{40}$$

It holds that $\left\|H_{f,x,\vec{A}}^T\right\|_2 \leq \Delta$ where Δ is a parameter of the scheme.

The secret key for a predicate f is a trapdoor T_f for $[A_0 \mid A_f]$, whose GS-norm is $\rho = O(\sqrt{n\log q})$. The trapdoor is used to sample a matrix $R \in \mathbb{Z}_q^{2m\times m}$ such that $[A_0 \mid A_f]R = D$ and $\|R^T\|_2 < 2m\rho\sigma$ for σ that is a parameter of the scheme. Decryption is computed by

$$\begin{aligned} c_{\text{out}}^T - \left[c_0^T \mid c_f^T\right] R = \quad & s^T D + e_{\text{out}}^T + \lceil q/2\rceil M^T \\ & - \left[s^T A_0 + e_0^T \mid s^T A_f + \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\vec{A}}\right] R \\ = \quad & \lceil q/2\rceil M^T + e_{\text{out}}^T - \left[e_0^T \mid \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\vec{A}}\right] R \end{aligned} \tag{41}$$

We observe that the largest coordinate of the noise vector

$$e_{\text{out}}^T - \left[e_0^T \mid \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\vec{A}}\right] R$$

is bounded by $\chi_{\max} + 2m\rho\sigma k\Delta\chi_{\max}$. The parameters of the scheme are chosen such that this bound is small enough compared to $q/4$, so one can round the result to extract M.

4.3 Applying Theorem 3.1

First, following Remark 2.21, the parameters of the scheme can be chosen to satisfy pairwise friendliness. Second, we claim that there exist an alternative decryption procedure, denote Decrypt', and a procedure SKCheck, such that the ABE scheme equipped with (SKCheck, Decrypt') satisfies the checkability property. Proving this claim is sufficient to apply our theorem to the scheme equipped with (SKCheck, Decrypt'), and by Remark 2.18 we conclude that the original scheme cannot be proved adaptively secure as well.

The implementation of SKCheck is relatively straightforward: Given the public parameters, a secret key SK, and a predicate f, verify that SK is a basis for $\Lambda_q^\perp([A_0 \mid A_f])$ and that its GS-norm is smaller than ρ. By definition of the scheme, we immediately get that any honestly-generated secret key passes the check, as desired.

Our strategy for the alternative decryption procedure is the following: Let SK_f be a (valid) secret key for policy f and let CT_x be a ciphertext such that $f(x) = 0$. Note that

$$A_x \begin{bmatrix} I & 0 \\ 0 & H_{f,x,\vec{A}} \end{bmatrix} = [A_0 \mid A_f] \tag{42}$$

So by Lemma 4.5, the secret key, which is a trapdoor T_f for $[A_0 \mid A_f]$, can be used to obtain a trapdoor T_x such that $\|T_x\|_{\mathrm{GS}} \leq \Delta\rho$. We use the "secret key" $\mathrm{SK}_x = T_x$ for A_x to "decode" and decrypt the ciphertext CT_x according to the following procedure. The procedure takes in a ciphertext, a secret key and public parameters.

Decode$(\mathrm{CT}_x, \mathrm{SK}_x, \mathrm{PP})$: Denote the components of the ciphertext by $c_0, \ldots, c_k$, c_{out} as defined in (38). Denote ρ' the GS-norm of SK_x. Use $T_x = \mathrm{SK}_x$ to obtain a matrix $R \in \mathbb{Z}^{m(k+1)\times m}$ such that $A_x R = D$ and $\|R^T\|_2 \le (k+1)m\rho'\sigma$. Compute $c_{\text{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R$ and round the result to extract $M \in \{0,1\}^m$ such that

$$\lceil q/2 \rceil M^T = \text{round}\left(c_{\text{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R\right) \tag{43}$$

Use T_x again to obtain a basis R' for $\Lambda_q^\perp([A_x \mid D])$ such that $\|R'\|_{\text{GS}} \le \rho'$. Compute the vector

$$y^T = \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\text{out}}^T - \lceil q/2 \rceil M^T\right] R' \tag{44}$$

Lift y to its canonical representative $\tilde{y} \in \left[-\frac{q}{2}, \frac{q}{2}\right)^{m(k+2)}$, compute R'^{-1} over the rationals, and compute $z^T = \tilde{y}^T R'^{-1}$. Let $z_0, z_1, \ldots, z_k, z_{\text{out}} \in \mathbb{Z}^m$ such that $z = (z_0, z_1, \ldots, z_k, z_{\text{out}})$. Check that the coordinates of $z_0, z_1, \ldots, z_k, z_{\text{out}}$ are smaller than $\chi_{\max}$. If not, output $\perp$, otherwise, output M.

We claim that the new procedure satisfies the ABE correctness requirement, that is, the result of the Decode procedure on honestly generated input is the message M encrypted in the input ciphertext. We prove this formally in the following lemma:

Lemma 4.6. For any $M \in \{0,1\}^m$, $x \in \{0,1\}^k$, honestly generated public parameters PP, honestly generated public key A_x for x, and honestly generated secret key T_f for for a predicate f such that $f(x) = 0$, it holds that

$$\text{Decrypt}(\text{Encrypt}(x, M, \text{PP}), T_x, \text{PP}) = M \tag{45}$$

Proof. Let CT_x be the result $\text{Encrypt}(x, M, \text{PP})$, then by definition of the scheme, CT_x is of the form

$$\begin{aligned} \mathrm{CT}_x^T = &\ \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\text{out}}^T\right] \\ = &\ s^T \left[A_0 \mid x_1 G + A_1 \mid \cdots \mid x_k G + A_k \mid D\right] \\ &+ \left[e_0^T \mid e_1^T \mid \cdots \mid e_k^T \mid e_{\text{out}}^T + \lceil q/2 \rceil M^T\right] \end{aligned} \tag{46}$$

where $s \xleftarrow{\$} \mathbb{Z}^n$ and $e_0, \ldots, e_k, e_{\text{out}} \leftarrow \chi^m$.

Since the secret key T_f is honestly generated, it satisfies $\|T_f\|_{\text{GS}} \le \rho$. As previously explained, T_f is used to obtain a trapdoor T_x for A_x that satisfies $\|T_x\|_{\text{GS}} \le \Delta\rho$.

Let R, R' be the low-norm matrices sampled in $\text{Decode}(\mathrm{CT}_x, T_x, \text{PP})$ using T_x, then $\|R^T\|_2 \le (k+1)m\Delta\rho\sigma$ and $\|R'\|_{\text{GS}} \le \Delta\rho$.

The first step of the decoding yields

$$\begin{aligned} c_{\text{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R &= \left(s^T D + \lceil q/2 \rceil M^T + e_{\text{out}}^T\right) - \left(s^T A_x + \left[e_0^T \mid \cdots \mid e_k^T\right]\right) R \\ &= \lceil q/2 \rceil M^T + \left(e_{\text{out}}^T - \left[e_0^T \mid \cdots \mid e_k^T\right] R\right) \end{aligned} \tag{47}$$

We observe that the largest coordinate of the noise vector $(e_{\text{out}}^T + \left[e_0^T \mid \cdots \mid e_k^T\right] R)$ is bounded by $\chi_{\max} + (k+1)m\rho\sigma\Delta\chi_{\max}$, so by the choice of the parameters of the scheme, this coordinate is small compared to $q/4$. Thus, the message extracted in Decode is M.

The second step of the decoding yields

$$y^T = \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\text{out}}^T - \lceil q/2 \rceil M^T\right] R' = \left[e_0^T \mid e_1^T \mid \cdots \mid e_k^T \mid e_{\text{out}}^T\right] R' \tag{48}$$

Lifting the result to $\tilde{y} \in \left[-\frac{q}{2}, \frac{q}{2}\right)^{m(k+2)}$ and computing $z^T = \tilde{y}^T R'^{-1}$ results in $z = (e_0, \ldots, e_k, e_{\text{out}})$. Since all $e_0, \ldots, e_k, e_{\text{out}}$ are sampled from $\chi_{\max}$-bounded distribution, the coordinates of z pass the check and the output of Decode is M, as desired. $\square$

Finally, we prove the following lemma to conclude the checkability of the alternative decryption:

Lemma 4.7. Let T_1, T_2 be two trapdoors for A_x such that $\|T_1\|_{\mathrm{GS}}, \|T_1\|_{\mathrm{GS}} \leq \Delta\rho$. For any ciphertext CT_x it holds that $\mathrm{Decode}(\mathrm{CT}_x, T_1, \mathrm{PP}) = \mathrm{Decode}(\mathrm{CT}_x, T_2, \mathrm{PP})$.

Proof. Denote $c_0, c_1, \ldots, c_k, c_{\mathrm{out}}$ as before. For $i = 1, 2$ let R_i, R_i' be the low-norm matrices sampled during $\mathrm{Decode}(\mathrm{CT}_x, T_i, \mathrm{PP})$ such that $A_x R_i = D$ and R_i' is low-norm basis for $\Lambda_q^{\perp}([A_x \mid D])$. If both decodings output $\perp$, the claim follows immediately. Therefore, assume w.l.o.g. that $\mathrm{Decode}(\mathrm{CT}_x, T_1, \mathrm{PP}) \neq \perp$. Let M be the binary vector extracted by $\mathrm{Decode}(\mathrm{CT}_x, T_1, \mathrm{PP})$, and let

$$y_1^T = \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\mathrm{out}}^T - \lceil q/2 \rfloor M^T\right] R_1' \tag{49}$$

Let $\tilde{y}_1 \in [-\frac{q}{2}, \frac{q}{2})^{m(k+2)}$ be the canonical representative of y_1, and let $R_1'^{-1}$ be the inverse of R_1' over the rationals. Let $z^T = \tilde{y}_1^T R_1^{-1}$ and $z_0, z_1, \ldots, z_k, z_{\mathrm{out}} \in \mathbb{Z}^m$ such that $z = (z_0, z_1, \ldots, z_k, z_{\mathrm{out}})$. By the assumption that $\mathrm{Decode}(\mathrm{CT}_x, T_1, \mathrm{PP}) \neq \perp$ we get that the coordinates of $z_0, z_1, \ldots, z_k, z_{\mathrm{out}}$ are smaller than $\chi_{\max}$. Since R_1' is a basis for $\Lambda_q^{\perp}([A_x|D])$ we conclude that there exists some vector $s \in \mathbb{Z}_q^n$ such that

$$\left[c_0^T \mid \cdots \mid c_k^T \mid c_{\mathrm{out}}^T - \lceil q/2 \rfloor M^T\right] = s^T [A_x \mid D] + \left[z_0^T \mid \cdots \mid z_k^T \mid z_{\mathrm{out}}^T\right] \tag{50}$$

We get that

$$\begin{aligned} c_{\mathrm{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R_2 &= \left(s^T D + \lceil q/2 \rfloor M^T + z_{\mathrm{out}}^T\right) - \left(s^T A_x + \left[z_0^T \mid \cdots \mid z_k^T\right]\right) R_2 \\ &= \lceil q/2 \rfloor M^T + \left(z_{\mathrm{out}}^T - \left[z_0^T \mid \cdots \mid z_k^T\right] R_2\right) \end{aligned} \tag{51}$$

By the requirement on the parameter of the scheme, the vector $z_{\mathrm{out}}^T - \left[z_0^T \mid \cdots \mid z_k^T\right] R_2$ is small enough compared to $q/4$, therefore the binary vector M' extracted by $\mathrm{Decode}(\mathrm{CT}_x, T_2, \mathrm{PP})$ satisfies

$$\lceil q/2 \rfloor M'^T = \mathrm{round}\left(c_{\mathrm{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R_2\right) = \lceil q/2 \rfloor M^T \tag{52}$$

i.e., $M' = M$. It remains to prove that $\mathrm{Decode}(\mathrm{CT}_x, T_2, \mathrm{PP})$ outputs the extracted vector M' and not $\perp$. We have

$$y_2^T = \left[c_0^T \mid \cdots \mid c_k^T \mid c_{\mathrm{out}}^T - \lceil q/2 \rfloor M'^T\right] R_2' = \left[z_0^T \mid \cdots \mid z_k^T \mid z_{\mathrm{out}}^T\right] R_2' \tag{53}$$

Lifting y_2 to its canonical representative and applying $R_2'^{-1}$ results in

$$z'^T = \tilde{y}_2^T R_2'^{-1} = \left[z_0^T \mid z_1^T \mid \cdots \mid z_k^T \mid z_{\mathrm{out}}^T\right] \tag{54}$$

Therefore, the checks on the coordinates of $z_0, \ldots, z_k, z_{\mathrm{out}}$ pass the same, and so

$$\mathrm{Decode}(\mathrm{CT}_x, T_1, \mathrm{PP}) = M = M' = \mathrm{Decode}(\mathrm{CT}_x, T_2, \mathrm{PP}) \tag{55}$$

□

References

[ABSV15] Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. From selective to adaptive security in functional encryption. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 657–677. Springer, 2015. doi:10.1007/978-3-662-48000-7_32.

[ABV+12] Shweta Agrawal, Xavier Boyen, Vinod Vaikuntanathan, Panagiotis Voulgaris, and Hoeteck Wee. Functional encryption for threshold functions (or fuzzy IBE) from lattices. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings*, volume 7293 of *Lecture Notes in Computer Science*, pages 280–297. Springer, 2012. doi:10.1007/978-3-642-30057-8_17.

[AFV11] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2011. doi:10.1007/978-3-642-25385-0_2.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 533–556, 2014.

[Boy13] Xavier Boyen. Attribute-based functional encryption on lattices. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 122–142. Springer, 2013. doi:10.1007/978-3-642-36594-2_8.

[BV16] Zvika Brakerski and Vinod Vaikuntanathan. Circuit-abe from LWE: unbounded attributes and semi-adaptive security. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 363–384, 2016.

[CGKW18] Jie Chen, Junqing Gong, Lucas Kowalczyk, and Hoeteck Wee. Unbounded ABE via bilinear entropy expansion, revisited. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 503–534. Springer, 2018. doi:10.1007/978-3-319-78381-9_19.

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer, 2003. doi:10.1007/3-540-39200-9_16.

[DDM15] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay. Compact attribute-based encryption and signcryption for general circuits from multilinear maps. In Alex Biryukov and Vipul Goyal, editors, *Progress in Cryptology - INDOCRYPT 2015 - 16th International Conference on Cryptology in India, Bangalore, India, December 6-9, 2015, Proceedings*, volume 9462 of *Lecture Notes in Computer Science*, pages 3–24. Springer, 2015. doi:10.1007/978-3-319-26617-6_1.

[DGP21] Cécile Delerablée, Lénaïck Gouriou, and David Pointcheval. Key-policy ABE with delegation of rights. *IACR Cryptol. ePrint Arch.*, page 867, 2021. URL: https://eprint.iacr.org/2021/867.

[DNS04] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *J. ACM*, 51(6):851–898, November 2004. doi:10.1145/1039488.1039489.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 40–49. IEEE Computer Society, 2013. Full version in [GGH+16]. doi:10.1109/FOCS.2013.13.

[GGH+16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.*, 45(3):882–929, 2016. URL: http://dx.doi.org/10.1137/14095772X, doi:10.1137/14095772X.

[GKW16] Rishab Goyal, Venkata Koppula, and Brent Waters. Semi-adaptive security and bundling functionalities made generic and easy. In Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 361–388, 2016.

[GKW17] Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 612–621. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.62.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*, pages 89–98. ACM, 2006. doi:10.1145/1180405.1180418.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 545–554. ACM, 2013. doi:10.1145/2488608.2488677.

[GW20] Junqing Gong and Hoeteck Wee. Adaptively secure ABE for DFA from k-lin and more. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 278–308. Springer, 2020. doi:10.1007/978-3-030-45727-3_10.

[IKOS08] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 433–442. ACM, 2008. doi:10.1145/1374376.1374438.

[KL15] Lucas Kowalczyk and Allison Bishop Lewko. Bilinear entropy expansion from the decisional linear assumption. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 524–541. Springer, 2015. doi:10.1007/978-3-662-48000-7_26.

[KW20] Lucas Kowalczyk and Hoeteck Wee. Compact adaptively secure ABE for NC1 from k-lin. *J. Cryptol.*, 33(3):954–1002, 2020. doi:10.1007/s00145-019-09335-x.

[LL20] Huijia Lin and Ji Luo. Compact adaptively secure ABE from k-lin: Beyond nc^1 and towards NL. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 247–277. Springer, 2020. doi:10.1007/978-3-030-45727-3_9.

[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, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2010. doi:10.1007/978-3-642-13190-5_4.

[LW12] Allison B. Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 180–198. Springer, 2012. doi:10.1007/978-3-642-32009-5_12.

[LW14] Allison B. Lewko and Brent Waters. Why proving HIBE systems secure is difficult. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 58–76. Springer, 2014. doi:10.1007/978-3-642-55220-5_4.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, 2012. doi:10.1007/978-3-642-29011-4_41.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 96–109. Springer, 2003. doi:10.1007/978-3-540-45146-4_6.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *Proceedings of the 2007 ACM Conference on*

Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007, pages 195–203. ACM, 2007. doi:10.1145/1315245.1315270.

[Pas11] Rafael Pass. Limits of provable security from standard assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 109–118. ACM, 2011. doi:10.1145/1993636.1993652.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93. ACM, 2005. doi:10.1145/1060590.1060603.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005. doi:10.1007/11426639_27.

[Tsa19] Rotem Tsabary. Fully secure attribute-based encryption for t-cnf from lwe. Cryptology ePrint Archive, Paper 2019/365, 2019. URL: https://eprint.iacr.org/2019/365.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009. doi:10.1007/978-3-642-03356-8_36.

[Wat11] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2011. doi:10.1007/978-3-642-19379-8_4.

[Wat15] Brent Waters. A punctured programming approach to adaptively secure functional encryption. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 678–697. Springer, 2015. doi:10.1007/978-3-662-48000-7_33.