Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-024-01358-1

Leakage-resilient IBE/ABE with optimal leakage rates from lattices

Qiqi Lai[1,2] · **Feng-Hao Liu**[3] · **Zhedong Wang**[2,4]

Received: 27 November 2022 / Revised: 8 January 2024 / Accepted: 11 January 2024

Abstract
We derive the first adaptively secure identity-based encryption (IBE) and attribute-based encryption (ABE) for t-conjunctive normal form formula (t-CNF), and selectively secure ABE for general circuits from lattices, with $1 - o(1)$ leakage rates, in the both relative leakage model and bounded retrieval model (BRM). To achieve this, we first identify a new fine-grained security notion for ABE—partially adaptive/selective security, and instantiate this notion from the learning with errors (LWE) assumption. Then, by using this notion, we design a new key compressing mechanism for identity-based/attributed-based weak hash proof system (IB/AB-wHPS) for various policy classes, achieving (1) succinct secret keys and (2) adaptive/selective security matching the existing non-leakage resilient lattice-based designs. Using the existing connection between weak hash proof system and leakage resilient encryption, the succinct-key IB/AB-wHPS can yield the desired leakage resilient IBE/ABE schemes with the optimal leakage rates in the relative leakage model. Finally, by further improving the prior analysis of the compatible locally computable extractors, we can achieve the optimal leakage rates in the BRM.

Keywords Leakage resilient · Optimal rate · Attribute-based · Lattice-based

Mathematics Subject Classification 94A60

Communicated by D. Stehle.

✉ Zhedong Wang
wzdstill@sjtu.edu.cn

Qiqi Lai
laiqq@snnu.edu.cn

Feng-Hao Liu
feng-hao.liu@wsu.edu

1 School of Computer Science, Shaanxi Normal University, Xi'an, China

2 State Key Laboratory of Cryptology, P. O. Box 5159, Beijing 100878, China

3 Washington State University, Pullman, WA, USA

4 School of Cyber Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Published online: 24 February 2024

1 Introduction

Leakage-resilient cryptography aims to create cryptosystems that maintain security even when partial information of the secret key is leaked. This line of study is motivated by both theoretical curiosities and perhaps more importantly, real-world scenarios, where some secure cryptosystems might be completely broken if some partial key information is given to the attackers. One famous example is the *side-channel attack* where the adversary can obtain leakage from measuring some physical behavior of an implementation [1, 28]. Another source of leakage comes from imperfect erasure where the attacker can obtain partial information before the content is completely erased, e.g., the *cold boot attacks* [25]. On the other hand, leakage resilience can be used to achieve security for other more complicated systems. For example, in the design of non-malleable codes, the works [21, 27, 32] leveraged leakage resilience to prove non-malleability. Therefore, leakage resilience has been an active research subject for the community, e.g., [4, 6, 7, 11, 20, 26, 34], to name a few.

1.1 Main goal

As motivated above, we aim to determine how to derive encryption schemes with better leakage rates, stronger security, and more expressive access control functionalities. More specifically, our goal is to construct leakage resilient encryption schemes in both the relative leakage model and the bounded retrieval model (BRM) with (1) optimal leakage rates, i.e., $1 - o(1)$, (2) post-quantum security and (3) more fine-grained access control, i.e., IBE and ABE for various classes of policy functions.

1.2 The leakage models

Various leakage models have been studied in the literature, capturing information leaked to the adversary. This work focuses on a simple yet general model called the *bounded-leakage model* (also known as the *memory leakage model*), allowing the attacker to learn arbitrary information about the secret key sk, as long as the number of leaked bits is bounded by some parameter ℓ. This model has drawn a lot of attentions [4, 6, 26, 34] for its elegance and simplicity, and can be used as a building block towards more sophisticated and realistic models, such as the continual leakage model [12, 19] (see [26]). Thus, understanding this model is not only of theoretical interest but also a necessary step towards realizing security for broader physical attacks.

The bounded leakage model would require $\ell < |\mathsf{sk}|$, as otherwise, the attacker can trivially obtain the whole secret key, and thus no meaningful security can be attained. To further characterize this requirement, there are two important models studied in the literature that treat the relation between ℓ and sk in a different way: (1) *relative leakage model*, and (2) *bounded retrieval model* (BRM).

In the former, the secret key and public-key are chosen in the same way as a standard cryptosystem (not necessary leakage resilient), and then the leakage parameter ℓ would be determined. The latter model generalizes the former by considering ℓ as an independent parameter whose growth (essentially) only goes with $|\mathsf{sk}|$, but would barely affect the other parameters, such as the public-key size, encryption running time, and ciphertext size. Basically, both models can scale up ℓ to allow an arbitrarily long leakage. But their difference is that the former would require to scale up the security parameter and thus all the other param-

eters, while the latter would only scale up the secret-key size and keep the other parameters essentially the same. Thus, constructions in the BRM is more desirable yet more challenging.

Leakage rate, i.e., the ratio $\ell/|\mathsf{sk}|$, is an important measure of efficiency for cryptosystems in these two models. Particularly, rate $1-o(1)$ is the best we can hope for—in order to tolerate ℓ bits of leakage, the system only needs to scale $|\mathsf{sk}|$ slightly larger than ℓ, optimizing the security/efficiency tradeoff.

1.3 Current state of the arts and challenges

We first notice that in the pre-quantum setting, leakage resilience can be achieved via the beautiful framework—*dual system encryption*, even for IBE/ABE and with optimal leakage rates [31]. However, current instantiations of the dual system encryption are all group-based [15, 22, 30, 31, 43, 44], and thus cannot defend against quantum algorithms. It is an interesting yet extremely challenging open question to instantiate dual system encryption from a post-quantum candidate, such as LWE or LPN.

For post-quantum leakage resilient encryption schemes, we notice that there are some limitations of the current techniques in achieving optimal leakage rate beyond the basic PKE. In prior work, there have been constructions of LWE/LPN-based PKE schemes with leakage rates $1-o(1)$, e.g., [14, 18], but their ideas do not generalize to more advanced settings, such as IBE and ABE. In a subsequent work, Hazay et al. [26] proposed a unified framework, showing that (1) PKE implies leakage resilient PKE in the relative leakage model, and (2) IBE implies leakage resilient PKE/IBE in the BRM. Moreover, the leakage resilient IBE achieves the same level of adaptive/selective security as that of the underlying IBE. Their idea can be generalized to construct leakage resilient ABE, but this approach inherently yields a very low leakage rate (i.e., $1/O(\kappa)$).

A recent work [36] somewhat mitigated this issue by improving the leakage rates, but at the cost of weaker security guarantees for the post-quantum instantiations. Particularly, they construct LWE-based leakage resilient IBE schemes in both the relative leakage model and the BRM, achieving $1-o(1)$ leakage rate in the former and $1-O(1)$ (for any arbitrarily small constant) in the latter. Their improvement relies on a novel *key-compression mechanism* that shortens the secret key length required in the framework of Hazay et al. [26]. Due to some technical limitation in the mechanism, their IBE scheme can only achieve the selective security. From these works [26, 36], we see a tradeoff between security and leakage rate, i.e., either we have an adaptively secure IBE with a low leakage rate, or a selectively secure IBE with a higher leakage rate.

1.4 Main question

In this work, we aim to further determine whether the tradeoff between (selective/adaptive) security and leakage rates as above is inherent. Particularly, we ask the following:

> *Can we achieve the optimal leakage rate* $(1-o(1))$ *for* IBE *(and* ABE *) in both relative and bounded retrieval models with security matching existing non-leakage resilient* IBE *(*ABE *), under* LWE?

1.5 Our contributions

In this work, we give positive answers in many settings of the main question. Our central idea is a refinement of the framework of [26, 36] by designing a new key compression mechanism for ABE *with succinct keys*. Below we describe our contributions in more detail.

- As a warm-up, we propose a new leakage model for ABE that incorporates parameters ℓ and ω, where ℓ is the number of bits allowed to leak per key and ω is the number of keys the adversary can leak. We note that for PKE and IBE, there is only one possible secret key corresponding to the public-key and the challenge id.[1] In this case, it is without loss of generality to just consider $\omega = 1$. However, for the ABE setting, there could be many possible secret keys corresponding to the challenge attribute, so specifying ω is natural and necessary in the leakage model. We call a scheme (ℓ, ω)-leakage resilient if the scheme can tolerate leakage on ω keys, each within ℓ bits.
- Next, we design improved instantiations of attribute-based weak hash proof system (AB-wHPS), which generalizes (identity-based) weak hash proof system [6, 26] by associating each ciphertext with an attribute and each secret key with a policy function. Particularly, we construct lattice-based AB-wHPS from ABE for various function classes, achieving two important new features: (1) succinct secret keys, i.e., the secret key length is $(|f| + o(|f|))$, where f is the policy function, and (2) security matching currently best known lattice-based ABE schemes (not necessarily leakage resilient). More specifically, we construct adaptively secure AB-wHPS for the class of comparison functions (which is the IB-wHPS) and the class t-CNF*,[2] and selectively secure AB-wHPS for general circuits.
- By using AB-wHPS for class $\mathcal{F}$ with *succinct keys*, we are able to construct $(\ell, 1)$-leakage resilient ABE for $\mathcal{F}$, with leakage rate $\ell/|\mathsf{sk}| = (1 - o(1))$ in the relative leakage model. We view AB-wHPS with succinct key as an improved key compression mechanism in comparison to prior works [26, 36] in the following two aspects: (1) AB-wHPS has better expressibility of policy function (the prior work [36] can only express the comparison function), and (2) we can derive adaptively secure AB-wHPS with succinct keys for classes which we have adaptively secure (non-leakage resilient) ABE. Prior to our work, for lattice-based schemes, we only had either a selectively secure IB-wHPS with succinct secret keys [36] or an adaptively secure IB-wHPS with non-succinct keys [26].
- From our AB-wHPS, we can further derive $(\ell, 1)$-leakage resilient ABE in the BRM, via an amplification and a connection with locally computable extractors as pointed out by [26]. However, prior compatible locally computable extractors [6] can only achieve $1 - O(1)$ leakage rate for an arbitrarily small constant. To achieve $1 - o(1)$ leakage rate, we improve the prior analysis [6] by refining their proof technique via the framework of Vadhan [41].
- Finally, we present a bootstrapping mechanism that generalizes our prior $(\ell, 1)$-leakage resilient ABE schemes to (ℓ, ω)-leakage resilient schemes for any bounded polynomial ω, in both relative leakage model and bounded retrieval model. The resulting leakage rate is still optimal (i.e., $1 - o(1)$) against block leakage functions, a slightly more restricted class.

[1] Here, we do not consider the case of randomizing the keys of PKE and IBE, as our main leakage-resilience results focus on bounded leakage case, rather than continual leakage case.

[2] This is the dual class for t-CNF used in [40]. Particularly, for t-CNF*, an assignment x is viewed as the policy function and the description of t-CNF is viewed as an attribute. In the general circuit model, the above reverse treatments are reasonable in theory. We use the dual class as we are working on key-policy ABE while the prior work [40] worked on ciphertext-policy ABE.

1.6 Overview of our techniques

Our central insight is a new key-compression mechanism for the framework in [26]. To illustrate our new idea, we first briefly review the prior framework [26] and point out the barrier in their leakage rates. Then we will describe our new ideas for the improvement.

1.6.1 (Weak) Hash proof system

A hash proof system can be described as a key encapsulation mechanism that consists of four algorithms (Setup, Encap, Encap^*, Decap): (1) Setup outputs a key pair $(\mathsf{pk}, \mathsf{sk})$, (2) $\mathsf{Encap}(\mathsf{pk})$ outputs a pair (CT, k) where k is a key encapsulated in a "valid" ciphertext CT, (3) $\mathsf{Encap}^*(\mathsf{pk})$ outputs an "invalid" ciphertext CT^*, and (4) $\mathsf{Decap}(\mathsf{sk}, \mathsf{CT})$ outputs a key k'. A (weak) hash proof system requires the following:

- **Correctness** For a valid ciphertext CT, Decap always outputs the encapsulated key $k' = k$, i.e., $\mathsf{Decap}(\mathsf{sk}, \mathsf{CT}) = k$, where $(\mathsf{CT}, k) \xleftarrow{\$} \mathsf{Encap}(\mathsf{pk})$.
- **Ciphertext indistinguishability** Valid ciphertexts and invalid ciphertexts are computationally indistinguishable, *even given the secret key*. This condition is essential for achieving leakage resilience [6, 34].
- **Universality** The decapsulation of an invalid ciphertext has information entropy, even for unbounded adversaries. Here, the randomness of invalid decapsulation comes from randomness in generating secret keys. A weak HPS (wHPS) only requires this property to hold for a random invalid ciphertext, i.e. $\mathsf{CT}^* \xleftarrow{\$} \mathsf{Encap}^*(\mathsf{pk})$, while a full-fledged HPS requires this to hold for any invalid ciphertext.

As noted in prior work [6], a wHPS already suffices to achieve leakage resilience, though it is not sufficient for CCA2 security, for which the design of HPS was originally intended [16]. Roughly speaking, the leakage resilient scheme derived from wHPS[6, 26, 34] can tolerate $\ell \approx |k| - \kappa$ bits of leakage, i.e., the length of encapsulated key minus security parameter, and thus the leakage rate of the derived encryption scheme would be $\ell/|\mathsf{wHPS.sk}| \approx \frac{|k|-\kappa}{|\mathsf{wHPS.sk}|}$.

Moreover, the idea can be generalized to IB-wHPS and AB-wHPS where an additional id or attribute $\boldsymbol{x}$ is associated with the ciphertext, and id or a policy function f is associated with the secret key. In the same way [26], IB-wHPS and AB-wHPS suffice to derive leakage resilient IBE and ABE.

1.6.2 wHPS from any PKE and generalizations [26]

While there were several instantiations of wHPS from specific assumptions [6, 34], Hazay et al. [26] showed somewhat surprisingly, any PKE implies wHPS. Their construction [26] can be broken into the following two steps: (1) construct a basic wHPS that only outputs 1 bit (or $\log \lambda$-bits), (2) amplify the output of the wHPS via parallel repetition. As pointed out in the work [26], parallel repetition might not amplify HPS in general, yet it does for wHPS as required in the application of leakage resilience.

The basic wHPS is simple: given any $\mathsf{PKE} = (\mathsf{Enc}, \mathsf{Dec})$, $\mathsf{wHPS.pk}$ consists of two public keys $\mathsf{pk}_0, \mathsf{pk}_1$ from PKE, and $\mathsf{wHPS.sk}$ is (b, sk_b) for a random bit b where sk_b corresponds to pk_b. The Encap algorithm outputs a valid ciphertext $\mathsf{CT} = (\mathsf{Enc}_{\mathsf{pk}_0}(k), \mathsf{Enc}_{\mathsf{pk}_1}(k))$ to encapsulate a uniformly random key $k \in \{0, 1\}$. The Encap^* algorithm outputs an invalid ciphertext $\mathsf{CT}^* = (\mathsf{Enc}_{\mathsf{pk}_0}(k), \mathsf{Enc}_{\mathsf{pk}_1}(1-k))$ for a uniformly random bit k. With n times parallel repetition, i.e., $\mathsf{wHPS}_{\|}.\mathsf{pk} := \{\mathsf{pk}_{i,0}, \mathsf{pk}_{i,1}\}_{i\in[n]}$ and $\mathsf{wHPS}_{\|}.\mathsf{sk} := \{(i, b_i), \mathsf{sk}_{i,b_i}\}_{i\in[n]}$,

we can get $\mathsf{wHPS}_{\|}$ with the encapsulated key $k = (k_i)_{i\in[n]}$ for an arbitrarily large $n \gg \kappa$, and thus a leakage resilient encryption that tolerates $\ell = (n - \kappa) \approx (n - o(|\mathsf{wHPS}_{\|}.\mathsf{sk}|))$.[3]

Naturally, this elegant approach can be generalized to construct IB-wHPS and AB-wHPS for class $\mathcal{F}$ from any IBE and ABE for $\mathcal{F}$, and the (adaptive/selective) security of the IB-wHPS and AB-wHPS matches the underlying IBE and ABE. Therefore, this framework provides a powerful way to design leakage resilient IBE and ABE from any IBE and ABE that can tolerate an arbitrarily large leakage ℓ.

1.6.3 Technical challenges from prior work

This technique of [26] achieves almost everything one would desire, except for the leakage rate. The main reason comes from the secret key size of $\mathsf{wHPS}_{\|}$, which is also scaled up by the parallel repetition, resulting in a low leakage rate as $\frac{\ell}{|\mathsf{wHPS}_{\|}.\mathsf{sk}|} = \frac{n-o(|\mathsf{wHPS}_{\|}.\mathsf{sk}|)}{|\mathsf{wHPS}_{\|}.\mathsf{sk}|} \approx \frac{n-o(n|\mathsf{PKE.sk}|)}{n|\mathsf{PKE.sk}|} \approx \frac{1}{|\mathsf{PKE.sk}|}$. To further improve the rate, it suffices to decrease $|\mathsf{wHPS.sk}|$ as observed by [36]. In particular, if we can shrink the secret key size of the wHPS to roughly $|\mathsf{wHPS}_{\|}.\mathsf{sk}| \approx n + |\mathsf{PKE.sk}|$, then the leakage rate would be $\frac{n-o(|\mathsf{wHPS}_{\|}.\mathsf{sk}|)}{|\mathsf{wHPS}_{\|}.\mathsf{sk}|} \approx \frac{n-o(n+|\mathsf{PKE.sk}|)}{n+|\mathsf{PKE.sk}|} \approx 1 - o(1)$, for sufficiently large n. Therefore, now the goal becomes to design a compact form of $\mathsf{wHPS}_{\|}.\mathsf{sk}$ that can encode n possible keys in a succinct way.

The work [36] achieved this goal and the more general IB-wHPS by proposing a novel key compression mechanism from a new primitive called *multi*-IBE. Then they instantiated the required multi-IBE from *inner-product encryption* (IPE) [2, 15, 44] with succinct keys. However, for lattice-based IPE schemes [2], only the selective security can be achieved under currently known techniques. Thus, the work [36] can only derive selectively secure leakage resilient IBE from lattices.

At this point, we summarize two limitations from the prior key compression mechanism [36]: (1) the approach is tied to IBE/IB-wHPS, and it is unclear whether we can further generalize the technique for further expressive policies, i.e., ABE; (2) the lattice-based instantiations are only selectively secure under currently known techniques. Below we show our new ideas to break these limitations.

1.6.4 Our new key compression mechanism

We first present a new key compression mechanism that can be generalized to more expressive policy functions, i.e., ABE. To illustrate our core insight, we first describe how to use the technique of key-policy (KP)-ABE to encode $\mathsf{wHPS}_{\|}.\mathsf{sk}$ succinctly. The idea can be naturally generalized to compress IB-wHPS and AB-wHPS. To facilitate further discussions, we first recall the concept of KP-ABE.

In a KP-ABE scheme, a secret key is associated with a policy function $f : \{0, 1\}^* \to \{0, 1\}$, and a ciphertext is associated with an attribute $\boldsymbol{x}$. The secret key can decrypt and recover the encrypted message if and only if $f(\boldsymbol{x}) = 1$.

Now we explain our key compression mechanism. Let us describe the format of a valid ciphertext of $\mathsf{wHPS}_{\|}$ as $\mathsf{CT} := \left\{\mathsf{Enc}_{\mathsf{pk}_{i,0}}(k_i), \mathsf{Enc}_{\mathsf{pk}_{i,1}}(k_i)\right\}_{i\in[n]}$, and a secret key is of the form

[3] Implicitly, we assume that the input min-entropy of an extractor is at least the security parameter κ. And such an extractor will be applied on the encapsulated key $k = (k_i)_{i\in[n]} \in \{0, 1\}^n$. Thus, the length of tolerated key-leakage should be $\ell = n - \kappa$. Moreover, it is implicitly assumed that $o(|\mathsf{wHPS}_{\|}.\mathsf{sk}|) \approx \kappa$, as $\mathsf{wHPS}_{\|}.\mathsf{sk}$ consists of a number of basic secret key $\mathsf{PKE.sk}$ and $\kappa = o(|\mathsf{PKE.sk}|)$.

$\{(i, b_i), \mathsf{sk}_{i,b_i}\}_{i\in[n]}$. From another angle looking at the ciphertext, we can view the indices (i, b)'s as attributes in an ABE, i.e. $\mathsf{CT} := \{\mathsf{ABE.Enc}(\mathsf{mpk}, (i, 0), k_i), \mathsf{ABE.Enc}(\mathsf{mpk}, (i, 1), k_i)\}_{i\in[n]}$. Then we can use a single ABE secret key to encode the set of keys $\{(i, b_i), \mathsf{sk}_{i,b_i}\}_{i\in[n]}$ as follows. Let $\boldsymbol{b} = (b_1, b_2, \ldots, b_n) \in \{0, 1\}^n$ be a binary vector, and define the following policy function $g_{\boldsymbol{b}}(i, z) = 1$ iff $b_i = z$ for each $i \in [n]$. In this way, only this set of attributes $\{(i, b_i)\}_{i\in[n]}$ satisfies the policy function $g_{\boldsymbol{b}}$, so the ABE decryption algorithm with $\mathsf{sk}_{g_{\boldsymbol{b}}}$ can successfully recover the encrypted messages from $\{\mathsf{ABE.Enc}(\mathsf{mpk}, (i, b_i), k_i)\}_{i\in[n]}$. The other part of the ciphertext, i.e., $\{\mathsf{ABE.Enc}(\mathsf{mpk}, (i, 1 - b_i), k_i)\}_{i\in[n]}$ is hidden by the security of the ABE. This approach can be naturally extended to the setting of IB-wHPS and AB-wHPS by adding an additional string $\boldsymbol{x} \in \{0, 1\}^*$ (either an ID or general attribute) to the existing attributes as above, resulting in ciphertexts of the form $\mathsf{CT} := \{\mathsf{ABE.Enc}(\mathsf{mpk}, (\boldsymbol{x}, i, 0), k_i), \mathsf{ABE.Enc}(\mathsf{mpk}, (\boldsymbol{x}, i, 1), k_i)\}_{i\in[n]}$. It is not hard to verify that these designs satisfy the requirements of (IB/AB)-wHPS.

Here we can conclude: (1) $\mathsf{sk}_{g_{\boldsymbol{b}}}$ is functionally equivalent to the set of secret keys $\{(i, b_i), \mathsf{sk}_{i,b_i}\}_{i\in[n]}$, and (2) as long as $\mathsf{sk}_{g_{\boldsymbol{b}}}$ has a succinct representation, i.e., $|\mathsf{sk}_{g_{\boldsymbol{b}}}|$ only depends on the depth but not the size of the function $g_{\boldsymbol{b}}$ when $g_{\boldsymbol{b}}$ is given, we can achieve the optimal leakage rate. We can instantiate the desired ABE by the lattice-based schemes [10, 23], and consequently derive a PKE/IBE/ABE with the optimal rate in the relative leakage model.

1.6.5 Adaptive security for various function classes

A careful reader may already observe that the underlying ABE schemes of [10, 23] do not achieve adaptive security, and neither do the IB-wHPS and AB-wHPS as constructed above. Moreover, it seems that lattice-based ABE that supports the computation of $g_{\boldsymbol{b}}(\cdot)$ with succinct keys (e.g., general circuits [10, 23]) can only achieve selective security. Thus, existing techniques plus the above approach do not suffice for our goal on adaptive security.

To overcome the limitation, we further observe that our constructions of IB-wHPS and AB-wHPS above actually *do not* require the full adaptive security of the whole attribute $(\boldsymbol{x}, (i, b))$ from the underlying ABE. We only need the selective security over the second part (i, b), as this part is generated by an honest key generation algorithm, instead of being challenged by the adversary.

With this insight, we define a more fine-grained security notion that considers partially adaptive/selective security over partitioned attributes $(\boldsymbol{x}, (i, b))$. Intuitively, if the underlying ABE is adaptively (or selectively) secure over $\boldsymbol{x}$ and *selective secure* over (i, b), then we can prove the AB-wHPS is adaptively (resp. selectively) secure. Furthermore we instantiate the required partially adaptive-selective ABE for various function classes. As a result, we obtain an adaptively secure IB-wHPS and AB-wHPS for t-CNF*, and selectively secure AB-wHPS for general circuits. This matches the function classes for which we know how to construct adaptively secure ABE without leakage.

1.6.6 Application

Our AB-wHPS with succinct keys immediately yields a $(\ell, 1)$-leakage resilient ABE with leakage rate $1 - o(1)$ in the relative leakage model, followed from the framework [26]. More specifically, by using our adaptively secure AB-wHPS for the comparison function (i.e., IB-wHPS) and the t-CNF* functions, we get leakage resilient and adaptively secure ABE for these classes with optimal leakage rates. Additionally, we can have selectively secure leakage resilient ABE for general circuits, with leakage rate $1 - o(1)$.

1.6.7 Extension I

As pointed out by [26], we can further derive $(\ell, 1)$-leakage resilient ABE in the BRM from AB-wHPS, via an amplification and a connection with locally computable extractors [41]. However, the analysis from prior compatible locally computable extractors only yields $1 - O(1)$ rate for the leakage resilient encryption scheme. It was left as an interesting open question by [36] how to improve the analysis of the extractor. We solve this open question by improving the analysis of the sampler [6] required by the general construction of Vadhan [41]. With our improved analysis, we are able to achieve $1 - o(1)$ leakage rate in the BRM.

1.6.8 Extension II

Finally, we show how to derive (ℓ, ω)-leakage resilient ABE with the optimal leakage rate in the block leakage setting for both relative model and BRM, for any bounded polynomial ω. Inspired by the work [24], we derive a new bootstrapping mechanism by connecting secret sharing with our AB-wHPS. We leave it as an interesting open question to achieve leakage resilient ABE even for an unbounded polynomial ω.

1.7 Other related work

AB-wHPS has been studied to construct leakage resilient ABE schemes in [45, 46]. Particularly, in [45], the authors focus on AB-wHPS supporting linear secret sharing schemes as the policy function class, from the pre-quantum decisional bilinear Diffie-Hellman assumption. The work in [46] constructed an AB-wHPS from a post-quantum, i.e, LWE, assumption. However, the constructions only achieve selective security for linear secret sharing schemes. And both of these related works only consider security in the relative leakage model. Compared with the prior works, our design/analysis approach is more modular, supporting broader function classes and/or stronger (adaptive) security.

1.8 Comparisons with the prior version in PKC 2022

In this section, we highlight the new contributions of the current paper, beyond the the prior version [29] published in PKC 2022. Generally, the current one is the full version, in which we add many more closely related background notions, definitions, and lemmas. All these are significantly helpful for unfamiliar but interested readers to understand our paper and this research field.

Besides, we supply the concrete partial-adaptively secure LWE-based ABE schemes and the detailed proofs for all theorems in the whole paper. They are quite important, as these details indicate that our new framework for optimal leakage-resilient encryption schemes can be concretely instantiated, instead of just a theoretically feasible result. Moreover, we believe that all these new added concrete ABE constructions are of independent interests, since its underlying key compression mechanism might inspire the researches in other settings related to the compactness of secret key.

Furthermore, from the view of techniques, we prove several interesting and necessary lemmas used for the parameter setting in pure mathematical way. And we believe these should be the first time that all of them are formally published by a journal or conference paper. And these lemmas can be used in other research fields.

Below, for clarity, we list all above mentioned new contents as follows.

- In the beginning of Sect. 2, we first add the used notations of this paper. Then, we present the necessary lattice background and technical tools for security proof in Sects. 2.2 and 2.5, respectively.
- In Sect. 3.2, we present the detailed proof of Theorem 3.12, which formally prove the security of Construction 3.11.
- In Sect. 4, we present the concrete instantiations of AB-wHPS from lattices. Particularly, we instantiate two partial-adaptively secure ABE schemes in Sects. 4.1.1 and 4.2, together with their parameter settings and formal security proofs.
- In Sect. 5, we present the detailed proof of Theorem 5.2, which formally prove the security of Construction 5.1.
- In Sect. 6.1, we present the detailed proof of Theorem 6.3, which formally prove the security of Construction 6.2.
- In Sect. 6.2, we present the detailed proof of Claims 6.12 and 6.13.
- In Sect. 6.3, we present the detailed proof of Theorem 6.16, which formally prove the security of Construction 6.15.
- In Sect. 7, we first introduce a useful lemma, Lemma 7.1, which is the key principle for the parameter setting of our construction. Notice that this lemma has been previously given as Lemma C.1 in [24]. However, it seems that their proof has certain flaws. In this section, we prove it again in a much more formal way. Then, we present the detailed proof of Theorem 7.3, which formally prove the security of Construction 7.2.

Besides, in order to help the readers to understand our construction of AB-wHPS more clearly, we recall the previously known basic wHPS in Online Appendix A.1.

1.9 Reading map

We first present the necessary mathematical notations, cryptographical definitions and related lemmas in Sect. 2. In Sect. 3, we introduce the notion of *attribute-based* weak hash proof system (AB-wHPS) and its general construction from ABE with a fine-grained security notion. In Sect. 4, we instantiate the AB-wHPS with three different function classes, through using or constructing three different ABE schemes. In Sect. 5, we show how to achieve the leakage resilient ABE with optimal leakage rate in the relative leakage model. Then, in Sect. 6, we enhance the schemes in Sect. 5 to achieve optimal leakage rate in the bounded retrieval model. Finally, in Sect. 7, we extend all above mentioned schemes to achieve leakage resilience in the multiple-key setting where the attacker can obtain leakage on ω possible decrypting keys for any bounded polynomial ω.

2 Preliminaries

In this section, we first introduce several standard mathematical notations for our constructions, then present necessary definitions and related lemmas.

2.1 Notations

In this paper, $\mathbb{Z}$ denotes the set of integers. We use κ to denote the security parameter, which is the implicit input for all algorithms presented in this paper. A function $f(\kappa) > 0$ is negligible and denoted by $\mathsf{negl}(\kappa)$ if for any $c > 0$ and sufficiently large κ, $f(\kappa) < 1/\kappa^c$. A probability

is said to be overwhelming if it is $1 - \mathsf{negl}(\kappa)$. A column vector is denoted by a bold lower case letter (e.g., $\boldsymbol{x}$). A matrix is denoted by a bold upright upper case letter (e.g., $\mathbf{A}$). For a vector $\boldsymbol{x}$, its Euclidean norm (also known as the ℓ_2 norm) is defined to be $\|\boldsymbol{x}\| = (\sum_i x_i^2)^{1/2}$. For a matrix $\mathbf{A}$, its ith column vector is denoted by $\boldsymbol{a}_i$ and its transposition is denoted by $\mathbf{A}^\top$. The Euclidean norm of a matrix is the norm of its longest column: $\|\mathbf{A}\| = \max_i \|\boldsymbol{a}_i\|$.

For a set D, we denote by $u \xleftarrow{\$} D$ the operation of sampling a uniformly random element u from D, and represent the bit length of u as $|u|$. For an integer $\ell \in \mathbb{N}$, we use U_ℓ to denote the uniform distribution over $\{0,1\}^\ell$. Given a randomized algorithm or function $f(\cdot)$, we use $y \xleftarrow{\$} f(x)$ to denote y as the output of f on input x. For a distribution X, we denote by $x \xleftarrow{\$} X$ the operation of sampling a random x according to the distribution X. Given two different distributions X and Y over a countable domain D, we can define their statistical distance to be $\Delta(X,Y) = \frac{1}{2}\sum_{d\in D} |X(d) - Y(d)|$, and say that X and Y are $\Delta(X,Y)$-close. Moreover, if $\Delta(X,Y)$ is negligible in κ, we say that the two distributions are statistically close, which is always denoted by $X \overset{s}{\approx} Y$. For any PPT algorithm $\mathcal{A}$, if $|\Pr[\mathcal{A}(1^\kappa, X) = 1] - \Pr[\mathcal{A}(1^\kappa, Y) = 1]|$ is negligible in κ, then we say that the two distributions are computationally indistinguishable, denoted by $X \overset{c}{\approx} Y$.

2.2 Lattices background

A lattice is a discrete additive subgroup of $\mathbb{R}^m$. Let $\mathbf{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m) \subset \mathbb{R}^m$ consist of m linearly independent vectors. The m-dimensional lattice Λ generated by the basis $\mathbf{B}$ is $\Lambda = \mathcal{L}(\mathbf{B}) = \{\mathbf{B}\cdot\boldsymbol{c} = \sum_{i\in[m]} c_i \cdot \boldsymbol{b}_i : \boldsymbol{c} = (c_1, \ldots, c_m) \in \mathbb{Z}^m\}$.

The minimum distance $\lambda_1(\Lambda)$ of a lattice Λ is the length in the Euclidean ℓ_2-norm of the shortest nonzero vector: $\lambda_1(\Lambda) = \min_{0\neq\mathbf{x}\in\Lambda} \|\boldsymbol{x}\|$. For an approximation factor $\gamma = \gamma(m) > 1$, we define the problem GapSVP$_\gamma$ as follows: given a basis $\mathbf{B}$ of an m-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$ and a positive number d, distinguish between the case where $\lambda_1(\Lambda) \le d$ and the case where $\lambda_1(\Lambda) \ge \gamma d$. We let $\widetilde{\mathbf{B}}$ denote the Gram-Schmidt orthogonalization of $\mathbf{B}$, and $\|\widetilde{\mathbf{B}}\|$ is the length of the longest vector in it.

In this paper, we will focus on a particular family of integer lattices. Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ for three positive integers m, n, q, where m and q are functions of n.[4] We consider the following two kinds of full-rank m-dimensional integer lattices defined by $\Lambda_q^\perp(\mathbf{A}) = \{\boldsymbol{e} \in \mathbb{Z}^m : \mathbf{A}\cdot\boldsymbol{e} = 0 \bmod q\}$ and its shift $\Lambda_q^{\boldsymbol{u}}(\mathbf{A}) = \{\boldsymbol{e} \in \mathbb{Z}^m : \mathbf{A}\cdot\boldsymbol{e} = \boldsymbol{u} \bmod q\}$.

Lemma 2.1 [5] *For any integers $n \ge 1$, $q \ge 2$, and sufficiently large $m = \lceil 6n\log q\rceil$, there is a probabilistic polynomial-time algorithm* $\mathsf{TrapGen}(q,n)$ *that outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m\times m})$ such that the distribution of $\mathbf{A}$ is statistically close to the uniform distribution over $\mathbb{Z}_q^{n\times m}$ and $\mathbf{T}_\mathbf{A}$ is a short basis for $\Lambda_q^\perp(\mathbf{A})$ satisfying $\|\widetilde{\mathbf{T}}_\mathbf{A}\| \le O(\sqrt{n\log q})$ and $\|\mathbf{T}_\mathbf{A}\| \le O(n\log q)$ with overwhelming probability.*

2.2.1 Gaussians on lattices

Let σ be any positive real number. The Gaussian distribution $\mathcal{D}_{\sigma,\boldsymbol{c}}$ with parameter σ and $\boldsymbol{c}$ is defined by probability distribution function $\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) = exp(-\pi\|\boldsymbol{x}-\boldsymbol{c}\|^2/\sigma^2)$. For any discrete and countable set $S \subseteq \mathbb{R}^m$, define $\rho_{\sigma,\boldsymbol{c}}(S) = \sum_{\boldsymbol{x}\in S}\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})$. The discrete Gaussian

[4] For certain applications, q should be prime.

distribution $D_{S,\sigma,\boldsymbol{c}}$ over S with parameters σ and $\boldsymbol{c}$ is defined by the probability distribution function $\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) = \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})/\rho_{\sigma,\boldsymbol{c}}(S)$ for all $\boldsymbol{x} \in S$.

Lemma 2.2 [3, Lemma 8] *Let* $\mathbf{A}$ *and* $\mathbf{T_A}$ *be a pair of matrices output by* $\mathsf{TrapGen}(q, n)$*, and* $r \geq \|\widetilde{\mathbf{T}}_{\mathbf{A}}\| \cdot \omega(\sqrt{\log m})$*. Then for* $\boldsymbol{c} \in \mathbb{R}^m$ *and* $\boldsymbol{u} \in \mathbb{Z}_q^n$*, we have:*

1. $\mathsf{Pr}[\boldsymbol{x} \leftarrow D_{\Lambda_q^{\boldsymbol{u}}(\mathbf{A}),r} : \|\boldsymbol{x}\| > r\sqrt{m}] \leq \mathsf{negl}(n)$.
2. *There is a probabilistic polynomial-time algorithm* $\mathsf{SampleGaussian}(\mathbf{A}, \mathbf{T_A}, r, \boldsymbol{c})$ *that outputs a sample from a distribution statistically close to* $D_{\Lambda,r,\boldsymbol{c}}$.
3. *There is a probabilistic polynomial-time algorithm* $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \boldsymbol{u}, r)$ *that outputs a sample from a distribution statistically close to* $D_{\Lambda_q^{\boldsymbol{u}}(\mathbf{A}),r}$.

The next two efficient algorithms $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ is used to generate identity secret key and prove anonymous indistinguishability for our new constructions.

Lemma 2.3 [3] *Given integers* $n \geq 1, q \geq 2$*, there exists some* $m = m(n, q) = O(n \log q)$*, there are sampling algorithms as follows:*

- *There is a* PPT *algorithm* $\mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}, \mathbf{T_A}, \boldsymbol{u}, s)$*, that takes as input: (1) a rank-n matrix* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$*, and any matrix* $\mathbf{B} \in \mathbb{Z}_q^{n\times m_1}$*, (2) a "short" basis* $\mathbf{T_A}$ *for lattice* $\Lambda_q^{\perp}(\mathbf{A})$*, a vector* $\boldsymbol{u} \in \mathbb{Z}_q^n$*, (3) a Gaussian parameter* $s > \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log(m + m_1)})$*; then outputs a vector* $\boldsymbol{r} \in \mathbb{Z}^{m+m_1}$ *distributed statistically close to* $\mathcal{D}_{\Lambda_q^{\boldsymbol{u}}(\mathbf{F}),s}$ *where* $\mathbf{F} := [\mathbf{A}|\mathbf{B}] \in \mathbb{Z}_q^{n\times(m+m_1)}$ *is an extension of* $\mathbf{A}$ *with* $\mathbf{B}$.
- *There is a* PPT *algorithm* $\mathsf{SampleRight}(\mathbf{A}, \mathbf{B}, \mathbf{R}, \mathbf{T_B}, \boldsymbol{u}, s)$*, that takes as input: (1) a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$*, and a rank-n matrix* $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$*, a matrix* $\mathbf{R} \in \mathbb{Z}_q^{m\times m}$*, where* $s_{\mathbf{R}} := \|\mathbf{R}\| = \sup_{\boldsymbol{x}:\|\boldsymbol{x}\|=1} \|\mathbf{R}\boldsymbol{x}\|$*, (2) a "short" basis* $\mathbf{T_B}$ *for lattice* $\Lambda_q^{\perp}(\mathbf{B})$*, a vector* $\boldsymbol{u} \in \mathbb{Z}_q^n$*, (3) a Gaussian parameter* $s > \|\widetilde{\mathbf{T}_{\mathbf{B}}}\| \cdot s_{\mathbf{R}} \cdot \omega(\sqrt{\log m})$*; then outputs a vector* $\boldsymbol{r} \in \mathbb{Z}^{2m}$ *distributed statistically close to* $\mathcal{D}_{\Lambda_q^{\boldsymbol{u}}(\mathbf{F}),s}$ *where* $\mathbf{F} := [\mathbf{A}|(\mathbf{AR} + \mathbf{B})] \in \mathbb{Z}_q^{n\times 2m}$.

2.2.2 Gadget matrix

We recall the "gadget matrix" $\mathbf{G}$ defined in [33]. The "gadget matrix" $\mathbf{G} = \mathbf{I}_n \otimes \boldsymbol{g}^{\top} \in \mathbb{Z}_q^{n\times n\lceil \log q\rceil}$ where $\boldsymbol{g}^{\top} = (1, 2, 4, \ldots, 2^{\lceil \log q\rceil - 1})$.

Lemma 2.4 [33, Theorem 1] *Let* q *be a prime, and* n, m *be integers with* $m = n\lceil \log q\rceil$*. There is a full-rank matrix* $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ *such that the lattice* $\Lambda_q^{\perp}(\mathbf{G})$ *has a publicly known trapdoor matrix* $\mathbf{T_G} \in \mathbb{Z}^{n\times m}$ *with* $\|\tilde{\mathbf{T}}_{\mathbf{G}}\| \leq \sqrt{5}$*, where* $\tilde{\mathbf{T}}_{\mathbf{G}}$ *is the Gram-Schmidt orthogonalization of* $\mathbf{T_G}$.

Lemma 2.5 [10, Lemma 2.1] *There is a deterministic algorithm, denoted by* $\mathbf{G}^{-1}(\cdot) : \mathbb{Z}_q^{n\times m} \to \mathbb{Z}^{m\times m}$*, that takes any matrix* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ *as input, and outputs the preimage* $\mathbf{G}^{-1}(\mathbf{A})$ *of* $\mathbf{A}$ *such that* $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A} \pmod q$ *and* $\|\mathbf{G}^{-1}(\mathbf{A})\| \leq m$.

Lemma 2.6 [3, Lemma 13] *Suppose that* $m > (n + 1)\log q + \omega(\log n)$ *and that* $q > 2$ *is a prime. let* $\mathbf{R}$ *be an* $m \times m$ *matrix chosen uniformly in* $\{0, 1\}^{m\times m}$*. Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be chosen uniformly in* $\mathbb{Z}_q^{n\times m}$*. Then for all vectors* $\boldsymbol{w} \in \mathbb{Z}_q^m$*, the distribution* $(\mathbf{A}, \mathbf{AR}, \mathbf{R}^{\top}\boldsymbol{w})$ *is statistically close to the distribution* $(\mathbf{A}, \mathbf{B}, \mathbf{R}^{\top}\boldsymbol{w})$.

2.2.3 Lattice homomorphic evaluation

We need to use the following homomorphic evaluation algorithms in [10].

Lemma 2.7 [10, 23] *Given integers $n > 1, q > 2$ and $m = O(n \log q)$, there exist three deterministic algorithms* $\mathsf{Eval}_{\mathsf{pk}}$,$\mathsf{Eval}_{ct}$ *and* Eval_{Sim} *as follows:*

- $\mathsf{Eval}_{\mathsf{pk}}(f, \mathbf{C}_1, \ldots, \mathbf{C}_\ell)$ *takes as input a d-depth circuit $f : \{0,1\}^\ell \to \{0,1\}$ and matrices $\mathbf{C}_1, \ldots, \mathbf{C}_\ell \in \mathbb{Z}_q^{n \times m}$, and outputs a matrix $\mathbf{C}_f \in \mathbb{Z}_q^{n \times m}$.*
- $\mathsf{Eval}_{ct}(f, \mathbf{C}_1, \ldots, \mathbf{C}_\ell, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_\ell, \boldsymbol{x})$ *takes as input a d-depth circuit $f : \{0,1\}^\ell \to \{0,1\}$, matrices $\mathbf{C}_i \in \mathbb{Z}_q^{n \times m}$, vectors $\boldsymbol{c}_i \in \mathbb{Z}_q^m$ and $\boldsymbol{x} \in \{0,1\}^\ell$, and outputs a vector $\boldsymbol{c}_f \in \mathbb{Z}_q^m$, such that if there exists some $\boldsymbol{s} \in \mathbb{Z}_q^n$ such that for every $i \in [\ell]$,*

$$\boldsymbol{c}_i = \boldsymbol{s}^\top (\mathbf{C}_i - x_i \mathbf{G}) + \boldsymbol{e}_i$$

with $\|\boldsymbol{e}_i\|_\infty \le B$, then

$$\boldsymbol{c}_f = \boldsymbol{s}^\top (\mathbf{C}_f - f(\boldsymbol{x})\mathbf{G}) + \boldsymbol{e}_f,$$

where $\|\boldsymbol{e}_f\|_\infty \le (m+1)^d \cdot B$.
- $\mathsf{Eval}_{Sim}(f, \{(x_i, \mathbf{R}_i)\}_{i=1}^\ell, \mathbf{A})$ *takes as input a d-depth circuit $f : \{0,1\}^\ell \to \{0,1\}$, $\boldsymbol{x} = (x_1, \cdots, x_\ell) \in \mathbb{Z}_q^\ell$, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{R}_1, \ldots, \mathbf{R}_\ell \in \{-1,1\}^{m \times m}$, and outputs a matrix $\mathbf{R}_f$ satisfying*

$$\mathbf{A}\mathbf{R}_f - f(\boldsymbol{x})\mathbf{G} = \mathbf{B}_f \text{ where } \mathbf{B}_f = \mathsf{Eval}_{\mathsf{pk}}(f, \mathbf{A}\mathbf{R}_1 - x_1\mathbf{G}, \ldots, \mathbf{A}\mathbf{R}_\ell - x_\ell\mathbf{G}),$$

and $\|\mathbf{R}_f\|_\infty \le 3 \cdot 4^d m + 1$

Furthermore, the running time of $\mathsf{Eval}_{\mathsf{pk}}$, Eval_{ct} *and* Eval_{Sim} *is $|f| \cdot \mathsf{poly}(n, \log q)$.*

We rely on the following lemma, which says that adding large noise "smudges" out any small values.

Lemma 2.8 (Smudging Lemma) *Let $B_1 = B_1(\kappa)$, and $B_2 = B_2(\kappa)$ be positive integers and let $e_1 \in [-B_1, B_1]$ be a fixed integer. Let $e_2 \leftarrow [-B_2, B_2]$ be chosen uniformly at random. Then the distribution of e_2 is statistically indistinguishable from that of $e_2 + e_1$ as long as $B_1/B_2 = \mathsf{negl}(\kappa)$.*

2.2.4 Learning with errors

The Learning with errors problem, or LWE, is the problem of determining a secret vector over $\mathbb{F}_q$ given a polynomial number of "noisy" inner products. The decision variant is to distinguish such samples from random. More formally, we define the problem as follows:

Definition 2.9 [38] Let $n \ge 1$ and $q \ge 2$ be integers, and let χ be a probability distribution on $\mathbb{Z}_q$. For $\boldsymbol{s} \in \mathbb{Z}_q^n$, let $A_{\boldsymbol{s},\chi}$ be the probability distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing a vector $\boldsymbol{a} \in \mathbb{Z}_q^n$ uniformly at random, choosing $e \in \mathbb{Z}_q$ according to χ and outputting $(\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e)$.

The decision $\mathsf{LWE}_{q,n,\chi}$ problem is: for uniformly random $\boldsymbol{s} \in \mathbb{Z}_q^n$, given a poly(n) number of samples that are either (all) from $A_{\boldsymbol{s},\chi}$ or (all) uniformly random in $\mathbb{Z}_q^n \times \mathbb{Z}_q$, output 0 if the former holds and 1 if the latter holds.

We say the decision-$\mathsf{LWE}_{q,n,\chi}$ problem is infeasible if for all probabilistic polyno- mial-time algorithms $\mathcal{A}$, the probability that $\mathcal{A}$ solves the decision-LWE problem (over $\boldsymbol{s}$ and $\mathcal{A}$'s random coins) is negligibly close to $1/2$ as a function of n. The works of [13, 37, 38] show that the LWE assumption is as hard as (quantum or classical) solving GapSVP and SIVP under various parameter regimes.

2.3 Attribute-based encryption (ABE)

Definition 2.10 (ABE [39]) An attribute-based encryption (ABE) scheme for a function class $\mathcal{F}_\kappa = \{f : \mathcal{X}_\kappa \to \{0,1\}\}$ consists of four algorithms $\mathsf{ABE}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ as follows.

- **Setup** $\mathsf{ABE.Setup}(1^\kappa)$ takes a security parameter κ as input, and generates a pair of master public key and master secret key $(\mathsf{mpk}, \mathsf{msk})$, where mpk contains the attribute space $\mathcal{X}_\kappa$, message space $\mathcal{M}$ and ciphertext space $\mathcal{CT}$.
- **Key generation** $\mathsf{ABE.KeyGen}(f, \mathsf{msk})$ takes as input a function (or circuit) $f \in \mathcal{F}_\kappa$ and the master secret key msk, and generates a secret key (f, sk_f). Without loss of generality, we think the secret key contains two parts, the function description f, and an extra sk_f. The secret key is succinct if $|\mathsf{sk}_f| = o(|f|)$.[5] When the context is clear, we often omit the description of f.
- **Encryption** $\mathsf{ABE.Enc}(\mathsf{mpk}, \boldsymbol{x}, \mu)$ takes as input the master public key mpk, an attribute $\boldsymbol{x} \in \mathcal{X}_\kappa$ and a message $\mu \in \mathcal{M}$, and outputs a ciphertext $\mathsf{ct} \in \mathcal{CT}$.
- **Decryption** $\mathsf{ABE.Dec}(\mathsf{sk}_f, \mathsf{ct})$ takes as input a secret key sk_f and a ciphertext c, and outputs $\mu \in \mathcal{M}$ if $f(\boldsymbol{x}) = 1$ and $\perp$ if $f(\boldsymbol{x}) = 0$, where $\boldsymbol{x}$ is the corresponding attribute used to generate ct.

Correctness We require that for all $f \in \mathcal{F}$, $\boldsymbol{x} \in \mathcal{X}_\kappa$, $\mu \in \mathcal{M}$, for correctly generated $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{ABE.Setup}(1^\kappa)$, $\mathsf{sk}_f \xleftarrow{\$} \mathsf{ABE.KeyGen}(\mathsf{msk}, f)$ and $\mathsf{ct} \xleftarrow{\$} \mathsf{ABE.Enc}(\mathsf{mpk}, \boldsymbol{x}, \mu)$, it holds that

- if $f(\boldsymbol{x}) = 1$, $\Pr\left[\mathsf{ABE.Dec}(\mathsf{sk}_f, \mathsf{ct}) = \mu\right] \geq 1 - \mathsf{negl}(\kappa)$.
- if $f(\boldsymbol{x}) = 0$, $\Pr\left[\mathsf{ABE.Dec}(\mathsf{sk}_f, \mathsf{ct}) = \perp\right] \geq 1 - \mathsf{negl}(\kappa)$.

2.3.1 Leakage resilience in the relative leakage model

Next, we give the formal definition of leakage-resilient key-policy ABE.

Definition 2.11 (*Leakage-resilient* ABE) A leakage-resilient ABE with attribute space $\mathcal{X}_\kappa$ for a class of functions $\mathcal{F}_\kappa = \{f : \mathcal{X}_\kappa \to \{0,1\}\}$ in the relative leakage model consists of four algorithms $\mathsf{ABE}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$, which are parameterized by a security parameter κ and leakage parameters ℓ, ω. In particular, (ℓ, ω)-leakage-resilient ABE can be defined by the following experiment.

We define the advantage of $\mathcal{A}$ in the above experiment[6] to be

$$\mathbf{Adv}^{\mathsf{LR}}_{\mathsf{ABE},\mathcal{A}}(\kappa, \ell, \omega) = \left|\Pr[b = b'] - 1/2\right|.$$

[5] For a general definition itself on ABE, there are not strict requirements on whether the size $|f|$ is fixed for all $f \in \mathcal{F}_\kappa$, and whether the size $|\mathsf{sk}_f|$ is independent of $|f|$. But for the instantiation of lattice-based ABE, we always set an upper bound for the circuit size $|f|$, and let $|\mathsf{sk}_f|$ depend on the depth of f, rather than $|f|$. Besides, with the consideration of leakage resilience, we assume that sk_f for arbitrary $f \in \mathcal{F}_\kappa$ can be encoded as bit-strings with fixed length.

[6] Notice that in the above experiment $\mathbf{Exp}^{\mathsf{LR}}_{\mathsf{ABE},\mathcal{A}}(\kappa, \ell, \omega)$, we allow the adversary to interleave key queries in *Test Stage 1* and leakage queries in *ω-Leakage queries Stage*, in an arbitrary way.

Experiment $\mathbf{Exp}^{\mathsf{LR}}_{\mathsf{ABE},\mathcal{A}}(\kappa, \ell, \omega)$

Attribute Challenge: In the selective setting, $\mathcal{A}$ chooses a challenge attribute $\boldsymbol{x}^* \in \mathcal{X}_\kappa$ before the Setup stage and sends it to $\mathcal{C}$; In the adaptive setting, $\mathcal{A}$ chooses an challenge $\boldsymbol{x}^* \in \mathcal{X}_\kappa$ in the challenge stage, and sends it to $\mathcal{C}$.
Test Stage 1: $\mathcal{A}$ adaptively queries the challenger $\mathcal{C}$ with function $f \in \mathcal{F}_\kappa$. For each query in the selective setting, $\mathcal{C}$ responds with (f, sk_f) if $f(\boldsymbol{x}^*) \neq 1$ and $\perp$ otherwise. ω-**Leakage Queries Stage:** $\mathcal{A}$ adaptively queries the challenger $\mathcal{C}$ with (f_i, h_i) for $i \in [\omega]$, where f_i is a policy function such that $f_i(\boldsymbol{x}^*) = 1$, and $h_i : \{0,1\}^{|\mathsf{sk}_{f_i}|} \to \{0,1\}^\ell$ is a leakage function. The adversary gets $h_i(\mathsf{sk}_{f_i})$ from $\mathcal{C}$.
Challenge Stage: $\mathcal{A}$ chooses two messages $\mu_0, \mu_1 \in \mathcal{M}$ and sends them to $\mathcal{C}$. Then $\mathcal{C}$ chooses $b \xleftarrow{\$} \{0,1\}$ and computes $\mathsf{ct}_b \xleftarrow{\$} \mathsf{ABE.Enc}(\mathsf{mpk}, \boldsymbol{x}^*, \mu_b)$. Finally, $\mathcal{C}$ returns ct_b to $\mathcal{A}$.
Test Stage 2: $\mathcal{A}$ adaptively queries the challenger $\mathcal{C}$ with function $f \in \mathcal{F}_\kappa$. Then in the selective setting, $\mathcal{C}$ responds with $(f, \mathsf{sk}_{\mathsf{id},f})$ if $f(\boldsymbol{x}^*) \neq 1$ and $\perp$ otherwise.
Output: The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

The scheme is (ℓ, ω)-leakage resilient if for any PPT adversary $\mathcal{A}$, we have $\mathbf{Adv}^{\mathsf{LR}}_{\mathsf{ABE},\mathcal{A}}(\kappa, \ell, \omega) \leq \mathsf{negl}(\kappa)$, and the leakage rate of this ABE is $\frac{\ell}{|\mathsf{sk}|}$.

Furthermore, the scheme is abbreviated as ℓ-leakage resilient if $\omega = 1$ in the above experiment.

Remark 2.12 We use the parameter ω to denote the number of different challenge keys on which leakage queries can be made. For PKE and IBE, we have $\omega = 1$ as for these two settings, there is a unique challenge key corresponding to the challenge attribute. For the more general ABE, there might be many different keys sk_{f_i} such that for the challenge attribute $\boldsymbol{x}^*$, $f_i(\boldsymbol{x}^*) = 1$. Thus, this parameter ω would be an important specification for the leakage resilient ABE.

Remark 2.13 In our security model, the adversary can obtain leakage on ω secret keys adaptively one after another. The secret keys would then form a block-source under the leakage.[7] We note that it is possible to generalize the model where the leakage function takes as inputs all the ω secret keys. In this work, we focus mainly on the block-source setting, as it already captures many useful scenarios.

2.3.2 Leakage resilience in the BRM

Below, we generalize to the setting of ABE the definition of leakage-resilience in the BRM by Alwen et al. [6].

Definition 2.14 (ABE *in the* BRM) An ABE for attribute space $\mathcal{X}_\kappa$ and policy function class $\mathcal{F} := \{\mathcal{X}_\kappa \to \{0,1\}\}$ is (ℓ, ω)-leakage resilient in the BRM if its master public-key size, ciphertext size, encryption time and decryption time (and the number of secret-key bits used by decryption) are independent of the leakage-bound ℓ. Besides, in the leakage resilient experiment, the adversary is allowed to conduct key leakage attacks on ω

[7] For the case that $\mathsf{sk} := S = (S_1, \ldots, S_m)$ is an $m \times e$ block source as in [42], we define leakage functions $f_i : \{0,1\}^* \to \{0,1\}^\ell$ independently for each block S_i with all $i \in [m]$. We say $(f_1, \ldots, f_m)$ are block leakage functions, if the min-entropy of S_i is still large enough even given leakage $(f_1(S_1), \ldots, f_{i-1}(S_{i-1}))$ for any $i \in [m]$. Clearly, when $m = 1$, this is the trivial case in Definition 2.11. Here, we call $\frac{m\ell}{|\mathsf{sk}|}$ the block leakage rate of the corresponding scheme.

secret keys corresponding to the challenge attribute. More formally, there exist polynomials $\mathsf{mpksize}$, ctsize, encT, decT, such that, for any polynomial ℓ and any $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{ABE.Setup}(1^\kappa, 1^{\ell(\kappa)})$, $\boldsymbol{x} \in \mathcal{X}_\kappa$, $\mu \in \mathcal{M}$, $\mathsf{ct} \xleftarrow{\$} \mathsf{ABE.Enc}(\mathsf{mpk}, \boldsymbol{x}, \mu)$, the scheme satisfies:

1. Master public-key size is $|\mathsf{mpk}| \leq O(\mathsf{mpksize}(\kappa))$, ciphertext size is $|\mathsf{ct}| \leq O(\mathsf{ctsize}(\kappa, |\mu|))$.
2. Run-time of $\mathsf{ABE.Enc}(\mu, \mathsf{pk})$ is bounded by $O(\mathsf{encT}(\kappa, |\mu|))$.
3. Run-time of $\mathsf{ABE.Dec}(\mathsf{ct}, \mathsf{sk}_f)$ and the number of bits of sk_f used in this decryption bounded by $O(\mathsf{decT}(\kappa, |\mu|))$, where $\mathsf{sk}_f \xleftarrow{\$} \mathsf{ABE.KeyGen}(\mathsf{msk}, f)$ with $f \in \mathcal{F}$ such that $f(\boldsymbol{x}) = 1$. Here we assume that the secret key sk_f is stored in a random access memory (RAM), and the decryption algorithm $\mathsf{ABE.Dec}(\mathsf{ct}, \cdot)$ only needs to read partial bits of sk_f to decrypt.

The leakage rate of this scheme is defined as $\frac{\ell}{|\mathsf{sk}_f|}$. Furthermore, the scheme is abbreviated as ℓ-leakage resilient if the parameter $\omega = 1$ in the experiment.

2.3.3 Policy function classes

This work considers three function classes: (1) ID comparison functions, (2) t-CNF^* formulas, and (3) general circuits. (1) and (3) are clear from the literature. We elaborate on (2). First we present the definition of the function class t-CNF.

Definition 2.15 (t-CNF [40]) A t-CNF policy $f : \{0,1\}^\ell \to \{0,1\}$ is a set of classes $f = \{(T_i, f_i)\}_i$, where for all i, $T_i \subseteq [\ell]$, $|T_i| = t$ and $f_i : \{0,1\}^t \to \{0,1\}$. For all $x \in \{0,1\}^\ell$ the value of $f(x)$ is computed as $f(x) = \bigwedge_i f_i(x_{T_i})$, where x_T is the length-t bit-string consisting of the bits of x in the indices T. A function class $\mathcal{F}$ is t-CNF if it consists only of t-CNF policies for some fixed $\ell \in \mathbb{N}$ and a constant $t \leq \ell$. If $\mathcal{F}$ is a t-CNF class, we say that t is the CNF locality of $\mathcal{F}$.

In this paper, we use the "dual" form of t-CNF, called t-CNF^*. The use of the dual version is because the prior work [40] worked on the ciphertext-policy ABE for t-CNF, and this work presents the result in the key-policy setting.

Definition 2.16 (t-CNF^*) For any $x \in \{0,1\}^\ell$ (the domain of t-CNF), let $U_x(\cdot)$ denote the function for which x is hardwired into $U_x(\cdot)$, and $U_x(\cdot)$ takes $f \in t$-CNF as input and outputs $U_x(f)$ such that $U_x(f) = f(x)$. $U_x(\cdot)$ is uniquely determined by x. We denote the function class $\{U_x(\cdot)\}$ as t-CNF^*.

2.4 Entropy and extractors

Definition 2.17 (*Min-entropy*) The min-entropy of a random variable X, denoted as $H_\infty(X)$ is defined as $H_\infty(x) = -\log\left(\max_{x_0 \in X} \Pr[x = x_0]\right)$.

Definition 2.18 (*Average-conditional min-entropy* [17]) The average-conditional min-entropy of a random variable X conditioned on a correlated variable Z, denoted as $H_\infty(X|Z)$ is defined as

$$H_\infty(X|Z) = -\log\left(\mathbb{E}_{z \leftarrow Z}[\max_x \Pr[X = x|Z = z]]\right) = -\log\left(\mathbb{E}_{z \leftarrow Z}[2^{H_\infty[X|Z=z]}]\right).$$

This notion of conditional min-entropy measures the best guess for X by an adversary that may observe an average-case correlated variable Z.

Lemma 2.19 [17] *Let X, Y, Z be arbitrarily correlated random variables where the support of Y has at most 2^ℓ elements. Then $H_\infty(X|(Y,Z)) \geq H_\infty(X|Z) - \ell$. In particular, $H_\infty(X|Y) \geq H_\infty(X) - \ell$.*

We also give the definition of randomness extractors [35], which is somewhat stronger than the average-case strong extractor [17].

Definition 2.20 (*Randomness extractor*) An efficient function $\mathsf{Ext} : \mathcal{X} \times \mathcal{S} \to \mathcal{Y}$ is a (v, ε)-extractor if for all (correlated) random variable X, Z such that the support of X is $\mathcal{X}$ and $H_\infty(X|Z) \geq v$, we have $\Delta((Z, S, \mathsf{Ext}(X; S)), (Z, S, Y)) \leq \varepsilon$, where S (also called the seed) and Y are distributed uniformly and independently over their domains $\mathcal{S}, \mathcal{Y}$ respectively.

Theorem 2.21 [17] *Let $\mathcal{H} = \{h_s : \mathcal{X} \to \mathcal{Y}\}_{s \in \mathcal{S}}$ be a universal family of hash functions meaning that for all $x = x' \in \mathcal{X}$ we have $\mathsf{Pr}_{s \leftarrow \mathcal{S}}[h_s(x) = h_s(x')] \leq \frac{1}{|\mathcal{Y}|}$. Then $\mathsf{Ext}(x, s) \stackrel{def}{=} h_s(x)$, is a (v, ε)-extractor for any parameter $v \geq \log|\mathcal{Y}| + 2\log(1/\varepsilon)$.*

2.5 Pairwise independent hash function

In order to prove the security of our concrete constructions, we need to use the partitioning strategy. As a preparation, we give a lemma which shows that pairwise independent hash function family which is denoted as $\mathcal{H}_{\mathsf{pind}}$ has the isolation property as long as a conditional probability defined as below approximates $1/|Q|$.

Lemma 2.22 ([8, Lemma 6.1]) *Let $Q \subseteq \{0,1\}^n$, A, B be integers such that $B \leq A$, $|Q| \leq \delta B$ for some $\delta \in (0,1)$, and let $\mathcal{H}_{\mathsf{pind}} : \{0,1\}^n \to \mathcal{Y}$ be an almost pairwise independent hash function family which has the following properties:*

- $\forall \boldsymbol{a} \in \{0,1\}^n$, $\mathsf{Pr}_{H \leftarrow \mathcal{H}_{\mathsf{pind}}}[H(\boldsymbol{a}) = 0] = 1/A$;
- $\forall \boldsymbol{a} \neq \boldsymbol{b} \in \{0,1\}^n$, $\mathsf{Pr}_{H \leftarrow \mathcal{H}_{\mathsf{pind}}}[H(\boldsymbol{a}) = 0 | H(\boldsymbol{b}) = 0] \leq 1/B$.

Then for any element $\boldsymbol{a} \notin Q$, we have

$$\mathsf{Pr}_{H \in \mathcal{H}_{\mathsf{pind}}}[H(\boldsymbol{a}) = 0 \bigwedge H(\boldsymbol{a}') \neq 0, \forall \boldsymbol{a}' \in Q] \in \left[\frac{1-\delta}{A}, \frac{1}{A}\right].$$

2.5.1 An explicit almost pairwise independent hash construction

Let $q \in \mathbb{N}$ be a prime, $t \in \mathbb{N}$, and let $f(x)$ be a monic irreducible polynomial in $\mathbb{Z}_q$ of degree $(t-1)$. Then we define $R = \mathbb{Z}_q[X]/\langle f(x)\rangle$, and note that R is isomorphic to $\mathbf{GF}(q^t)$ as q is a prime and $f(x)$ is an irreducible polynomial of degree $(t-1)$. We will use R as the representation of $\mathbf{GF}(q^t)$. We then define two mappings $\phi : R \to \mathbb{Z}_q^t$ and $\mathsf{Rot} : R \to \mathbb{Z}_q^{t \times t}$ by

$$\begin{aligned}
&\phi : \theta = a_1 + a_2 x + \cdots + a_t x^{t-1} \mapsto (a_1, \ldots, a_t)^\top, \\
&\mathsf{Rot} : \theta = a_1 + a_2 x + \cdots + a_t x^{t-1} \mapsto \left[\phi(\theta)\phi(\theta x) \ldots \phi(\theta x^{t-1})\right].
\end{aligned}$$

We note that $\mathsf{Rot}(\theta) \cdot \phi(\vartheta) = \phi(\theta\vartheta)$, $\mathsf{Rot}(\theta) \cdot \mathsf{Rot}(\vartheta) = \mathsf{Rot}(\theta\vartheta)$, and $\mathsf{Rot}(\theta) + \mathsf{Rot}(\vartheta) = \mathsf{Rot}(\theta + \vartheta)$. This means that Rot is a ring-homomorphism from R to $\mathbb{Z}_q^{t \times t}$. If $\theta \neq \theta' \in \mathbf{GF}(q^t)$, then $\mathsf{Rot}(\theta) - \mathsf{Rot}(\theta') = \mathsf{Rot}(\theta - \theta') \neq 0$.

For any $h \in \mathbf{GF}(q^t)$, we define $G(h)$ as $G(h) := \mathsf{Rot}(h) \in \mathbb{Z}_q^{t\times t}$, then we define an pairwise independent hash function family $\mathcal{H}_{\mathsf{pind}} : \mathbb{Z}_q^\ell \to \mathbb{Z}^{n\times n}$ where $t|n$ as: $\forall H \in \mathcal{H}_{\mathsf{pind}}$, H is indexed by $(h_1, \ldots, h_\ell) \in \mathbf{GF}(q^t)^\ell$, $\forall \boldsymbol{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}_q^\ell$, $H(\boldsymbol{x}) = \mathbf{I}_n + \sum_{i=1}^{\ell} x_i(G(h_i) \otimes \mathbf{I}_{n/t})$. We have the following lemma.

Lemma 2.23 [3, 8] *The function family $\mathcal{H}_{\mathsf{pind}}$ defined above is an pairwise independent hash function. Moreover, we have*

- $\forall H \leftarrow \mathcal{H}_{\mathsf{pind}}$ *and* $\forall \boldsymbol{a} \in \{0,1\}^\ell$, $\mathsf{Pr}[H(\boldsymbol{a}) = 0] = (1/q)^t$.
- $\forall H \leftarrow \mathcal{H}_{\mathsf{pind}}$ *and* $\forall \boldsymbol{a} \neq \boldsymbol{b} \in \{0,1\}^\ell$, $\mathsf{Pr}[H(\boldsymbol{b}) = 0 | H(\boldsymbol{a}) = 0] \leq (1/q)^t$.

3 Attribute-based weak hash proof systems

In this section, we first present a generalization of the weak hash proof system called *attribute-based* weak hash proof system (AB-wHPS). This notion associates attributes and policy functions to the system following the spirit of attribute-based encryption. Next, we show how to construct AB-wHPS from ABE that achieves the property of *succinct keys*, which is the key to leakage resilience with the optimal rate. With a new fine-grained approach, we are able to achieve AB-wHPS with selective security for general circuits, adaptive security of identity comparison functions (i.e., identity-based wHPS), and adaptive security for t-CNF* functions, from lattices. This would imply lattice-based leakage resilient, adaptively secure PKE, IBE, ABE for t-CNF*, and selectively secure ABE for general circuits, all with the optimal rate, matching the best known non-leakage resilient selectively/adaptively secure constructions.

3.1 Formal definition of attribute-based wHPS

We first present the formal definition of an AB-wHPS.

Definition 3.1 (AB-wHPS) An attribute-based weak hash proof system (AB-wHPS) for an attribute space $\mathcal{X}_\kappa = \{0,1\}^*$ and a class of functions $\mathcal{F}_\kappa = \{f : \mathcal{X}_\kappa \to \{0,1\}\}$ consists of five algorithms AB-wHPS.{Setup, KeyGen, Encap, Encap*, Decap}:

- **Setup** AB-wHPS.Setup(1^κ) takes a security parameter κ as input, and generates a pair of master public key and master secret key $(\mathsf{mpk}, \mathsf{msk})$. The attribute space $\mathcal{X}_\kappa$ and the encapsulated key space $\mathcal{K}$ are determined by mpk.
- **Key generation** AB-wHPS.KeyGen (f, msk) takes as input a function $f \in \mathcal{F}_\kappa$ and the master secret key msk, and generates a secret key (f, sk_f). Without loss of generality, we think the secret key contains two parts, the function description f, and an extra sk_f. The secret key is succinct if $|\mathsf{sk}_f| = o(|f|)$. When the context is clear, we often omit the description of f.
- **Valid encapsulation** AB-wHPS.Encap$(\mathsf{mpk}, \boldsymbol{x})$ takes as input the master public key mpk and an attribute $\boldsymbol{x} \in \mathcal{X}_\kappa$, and outputs a valid ciphertext CT and its corresponding encapsulated key $k \in \mathcal{K}$.
- **Invalid encapsulation** AB-wHPS.Encap$^*(\mathsf{mpk}, \boldsymbol{x})$ takes as input the master public key mpk and $\boldsymbol{x} \in \mathcal{X}_\kappa$, and outputs an invalid ciphertext CT^*.
- **Decapsulation** AB-wHPS.Decap$(\mathsf{sk}_f, \mathsf{CT})$ takes as input a secret key sk_f and a ciphertext CT, and deterministically outputs $k \in \mathcal{K}$ if $f(\boldsymbol{x}) = 1$ and $\perp$ if $f(\boldsymbol{x}) = 0$, where $\boldsymbol{x}$ is the corresponding attribute used to generate CT.

Table 1 Valid/invalid ciphertext indistinguishability experiment of AB-wHPS

Valid/Invalid Ciphertext Indistinguishability Experiment
Attribute Challenge: In the selective setting, $\mathcal{A}$ chooses an challenge attribute $\boldsymbol{x}^* \in \mathcal{X}_\kappa$ before the Setup stage and sends it to $\mathcal{C}$; In the adaptive setting, $\mathcal{A}$ chooses a challenge $\boldsymbol{x}^* \in \mathcal{X}_\kappa$ in any arbitrary stage before the challenge stage, and sends it to $\mathcal{C}$.
Setup: The challenger $\mathcal{C}$ gets a pair of $(\mathsf{mpk}, \mathsf{msk})$ by running $\mathsf{AB\text{-}wHPS.Setup}(1^\kappa)$, and sends mpk to $\mathcal{A}$.
Test Stage 1: $\mathcal{A}$ adaptively queries the challenger $\mathcal{C}$ with $f \in \mathcal{F}_\kappa$, and $\mathcal{C}$ responds with (f, sk_f).
Challenge Stage: $\mathcal{C}$ selects $b \xleftarrow{\$} \{0,1\}$. If $b = 0$, $\mathcal{C}$ computes $(\mathsf{CT}, k) \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x}^*)$. If $b = 1$, $\mathcal{C}$ computes $\mathsf{CT} \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x}^*)$. Then $\mathcal{C}$ returns CT to $\mathcal{A}$.
Test Stage 2: $\mathcal{A}$ adaptively queries the challenger $\mathcal{C}$ with $f \in \mathcal{F}$. Then $\mathcal{C}$ responds with (f, sk_f).
Output: $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$. $\mathcal{A}$ wins the experiment, if $b = b'$ and at most one of $\mathcal{A}$'s key queries f satisfies $f(\boldsymbol{x}^*) = 1$.

Furthermore, an AB-wHPS needs to satisfy three properties: correctness, ciphertext indistinguishability, and universality.

3.1.1 Correctness

For $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Setup}(\kappa)$, any $\boldsymbol{x} \in \mathcal{X}_\kappa$ and any $f \in \mathcal{F}_\kappa$ such that $f(\boldsymbol{x}) = 1$, we have

$$\Pr\Big[k = k' \Big| \mathsf{sk}_f \xleftarrow{\$} \mathsf{AB\text{-}wHPS.KeyGen}(f, \mathsf{msk}), (\mathsf{CT}, k) \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x}), k' = \mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, c)\Big] = 1.$$

3.1.2 Ciphertext indistinguishability

For any challenge attribute $\boldsymbol{x}^*$, valid/in-valid ciphertexts output by $\mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x}^*)$ and $\mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x}^*)$ are indistinguishable, even given one secret "1-key" sk_f such that $f(\boldsymbol{x}^*) = 1$ and perhaps many "0-keys" $\mathsf{sk}_{f'}$ such that $f'(\boldsymbol{x}^*) = 0$. More formally, this indistinguishability is always described by the experiment between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ in Table 1.

We define the advantage of $\mathcal{A}$ in the above game to be $\mathbf{Adv}^{\mathsf{AB\text{-}wHPS}}_{\Pi,\mathcal{A},\mathcal{F}_\kappa}(\kappa) = |\Pr[\mathcal{A}\ wins] - 1/2|$. The indistinguishability means that $\mathbf{Adv}^{\mathsf{AB\text{-}wHPS}}_{\Pi,\mathcal{A},\mathcal{F}_\kappa}(\kappa) \leq \mathsf{negl}(\kappa)$.

Remark 3.2 In this definition, we require ciphertext indistinguishability to hold even given a single sk_f such that $f(\boldsymbol{x}^*) = 1$. This suffices to achieve leakage resilient PKE, IBE, and $(\ell, 1)$-leakage resilient ABE directly, and (ℓ, ω)-leakage resilient ABE for any bounded-polynomial ω via a bootstrapping procedure (ref. Sect. 7), where $\ell \approx (1 - o(1))|\mathsf{sk}_f|$.

3.1.3 Universality

We need one additional information theoretic property, requiring that for any adversary with public parameters, the decapsulation of an invalid ciphertext has information entropy. We define this property as follow.

Definition 3.3 (*Universal* AB-wHPS) We say that an AB-wHPS is $(l, \bar{w})$-universal, if for any attribute $\boldsymbol{x} \in \mathcal{X}_\kappa$, $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Setup}(1^\kappa)$, and $\mathsf{CT}^* \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x})$, it holds

$$H_\infty(\mathsf{AB\text{-}wHPS.Decap}(\mathsf{CT}^*, \mathsf{sk}_f)|\mathsf{mpk}, \mathsf{msk}, \mathsf{CT}^*, \boldsymbol{x}) \geq \bar{w},$$

where $\mathsf{sk}_f = \mathsf{AB\text{-}wHPS.KeyGen}(f, \mathsf{msk})$ with $f(\boldsymbol{x}) = 1$, and l is the bit-length of the decapsulated value from $\mathsf{AB\text{-}wHPS.Decap}(\mathsf{CT}^*, \mathsf{sk})$.

3.2 Fine-grained security notions and general construction of AB-wHPS from ABE

In this section, we present how to construct AB-wHPS from ABE. To achieve adaptive security for several subclasses of policy functions, we present a more fine-grained approach as follows. We first define a notion called partially selective/adaptive security over partitioned attributes. Next we show for a *specific class* $\mathcal{G}$, if an ABE is $(\mathsf{X}, \mathsf{sel})$-secure for class $\mathcal{F} \wedge_\| \mathcal{G}$ with $\mathsf{X} \in \{\mathsf{sel}, \mathsf{ada}\}$,[8] then we can construct an X-secure AB-wHPS for $\mathcal{F}$. Moreover, suppose the underlying ABE has succinct keys, so does the AB-wHPS. In the next section, we show instantiations of $(\mathsf{ada}, \mathsf{sel})$-secure ABE for various function classes. Below we elaborate on the notations and the new security definition.

Definition 3.4 Let $\mathcal{F}_1 = \{f_1 : \mathcal{X}_1 \to \{0,1\}\}$ and $\mathcal{F}_2 = \{f_2 : \mathcal{X}_2 \to \{0,1\}\}$ be two function classes. We define the operator $\wedge_\|$ over two function classes as follow: $\mathcal{F} := \mathcal{F}_1 \wedge_\| \mathcal{F}_2$ is a function class that consists of function maps $\mathcal{X}_1 \times \mathcal{X}_2 \to \{0,1\}$, where each function $f_{f_1,f_2} \in \mathcal{F}$ is indexed by two functions $f_1 \in \mathcal{F}_1$ and $f_2 \in \mathcal{F}_2$ such that on input $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2) \in \mathcal{X}_1 \times \mathcal{X}_2$, $f_{f_1,f_2}(\boldsymbol{x}) = f_1(\boldsymbol{x}_1) \wedge f_2(\boldsymbol{x}_2)$.

Using this composed function class in Definition 3.4, we can naturally consider any combination of selective/adaptive security for ABE as follows.

Definition 3.5 (*Partial selective/adaptive security*) For any ABE with the attribute space $\mathcal{X}_1 \times \mathcal{X}_2$ for the policy function class $\mathcal{F} := \mathcal{F}_1 \wedge_\| \mathcal{F}_2$ defined as in Definition 3.4, we define partial selective/adaptive security as follows:

- ada-sel security: For any challenge attribute $\boldsymbol{x}^* = (\boldsymbol{x}_1^*, \boldsymbol{x}_2^*) \in \mathcal{X}_1 \times \mathcal{X}_2$, $\boldsymbol{x}_1^*$ is chosen adaptively but $\boldsymbol{x}_2^*$ is chosen selectively in the corresponding indistinguishability experiment.
- sel-ada security: For any challenge attribute $\boldsymbol{x}^* = (\boldsymbol{x}_1^*, \boldsymbol{x}_2^*) \in \mathcal{X}_1 \times \mathcal{X}_2$, $\boldsymbol{x}_1^*$ is chosen selectively and $\boldsymbol{x}_2^*$ is chosen adaptively in the corresponding indistinguishability experiment.

This notion also captures the standard selective (or adaptive) security as sel-sel (or ada-ada) security, where both parts of the challenge attribute are chosen selectively (or adaptively).

[8] The formal definition of $\mathcal{F} \wedge_\| \mathcal{G}$ is presented in the following Definition 3.4.

Remark 3.6 In this work, we need a slightly weaker version of the partial selective/adaptive security from ABE—the adversary is only allowed to query one key (f, g) such that $f(x_1^*) = 1$ and $g(x_2^*) = 0$. The other keys are of the form (f', g') such that $f'(x_1^*) = 0$. Therefore, throughout this work we will use this slightly weaker version by default.

Remark 3.7 In the same way, we can define the partial selective/adaptive ciphertext indistinguishability for AB-wHPS.

Remark 3.8 This definition can be defined recursively. For example, the first part $\mathcal{F}_1$ can also consists of two parts, i.e., $\mathcal{F}_1 = \mathcal{F}_{1,1} \wedge_{\|} \mathcal{F}_{1,2}$. In this case, we can consider (X-Y)-Z security for any combination of $\mathsf{X}, \mathsf{Y}, \mathsf{Z} \in \{\mathsf{sel}, \mathsf{ada}\}$.

To construct our desired AB-wHPS for $\mathcal{F}$, we need an ABE for $\mathcal{F} \wedge_{\|} \mathcal{G}$ for this specific $\mathcal{G}$ as we describe below.

Definition 3.9 Let $m = m(\kappa)$ and $n = n(\kappa)$ be two integer parameters, and we define a function class $\mathcal{G} = \{g : [n] \times [m] \to \{0, 1\}\}$ as follows. Each function $g_{\boldsymbol{y}} \in \mathcal{G}$ is indexed by a vector $\boldsymbol{y} = (y_1, \ldots, y_n)^\top \in [m]^n$, and $g_{\boldsymbol{y}}(x_1, x_2) = 1$ if and only if $x_2 = y_{x_1}$.

Remark 3.10 The class $\mathcal{G}$ can be captured by boolean circuits with input length $\log n + \log m$, and depth within $O(\log(n + m))$, i.e., $\bigvee_{i \in [n]} (i \stackrel{?}{=} x_1) \wedge (y_i \stackrel{?}{=} x_2)$.

Given this particular class $\mathcal{G}$ (with parameters m, n) defined in Definition 3.9 and a class $\mathcal{F}$, we show how to use ABE for $\mathcal{F} \wedge_{\|} \mathcal{G}$ to construct AB-wHPS for $\mathcal{F}$. For different classes $\mathcal{F}$'s, the AB-wHPS can be used to further derive leakage resilient PKE, IBE, and ABE.

Construction 3.11 (AB-wHPS from ABE) *Let* $\Pi_{\mathsf{ABE}} = \mathsf{ABE}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *be an* ABE *scheme with attribute-space* $\bar{\mathcal{X}}_\kappa = \mathcal{X}_\kappa \times \mathcal{X}'_\kappa = \{0, 1\}^* \times \{[n] \times [m]\}$*, message-space* $\mathcal{M} = \mathbb{Z}_m$ *and ciphertext space* $\mathcal{CT}$ *for the policy-function class* $\mathcal{F} \wedge_{\|} \mathcal{G}$ *for the class* $\mathcal{G}$ *as in Definition 3.9 with parameters* m, n*. Then, an* AB-wHPS $\Pi_{\mathsf{AB\text{-}wHPS}}$ *with attribute space* $\mathcal{X}_\kappa = \{0, 1\}^*$ *and the encapsulated-key-space* $\mathcal{K} = \mathbb{Z}_m^n$ *for the policy-function class* $\mathcal{F} = \{f : \{0, 1\}^* \to \{0, 1\}\}$ *can be constructed as follows:*

- $\mathsf{AB\text{-}wHPS.Setup}(1^\kappa)$*: Given the security parameter* κ *as input, the algorithm runs* $\mathsf{ABE.Setup}$ *to generate* $(\mathsf{mpk}^{\mathsf{ABE}}, \mathsf{msk}^{\mathsf{ABE}}) \stackrel{\$}{\leftarrow} \mathsf{ABE.Setup}(1^\kappa)$*, and outputs* $\mathsf{mpk} := \mathsf{mpk}^{\mathsf{ABE}}$ *and* $\mathsf{msk} := \mathsf{msk}^{\mathsf{ABE}}$.
- $\mathsf{AB\text{-}wHPS.KeyGen}(\mathsf{msk}, f)$*: Given a master secret-key* $\mathsf{msk} := \mathsf{msk}^{\mathsf{ABE}}$ *and a function* $f \in \mathcal{F}$ *as input, the algorithm first chooses a random vector* $\boldsymbol{y} \stackrel{\$}{\leftarrow} [m]^n$*, and sets* $\hat{f} := \hat{f}_{f, g_{\boldsymbol{y}}} \in \mathcal{F} \wedge_{\|} \mathcal{G}$*. Then the algorithm runs* $\mathsf{ABE.KeyGen}$ *to generate* $\mathsf{sk}_{\hat{f}}^{\mathsf{ABE}} \stackrel{\$}{\leftarrow} \mathsf{ABE.KeyGen}(\mathsf{msk}^{\mathsf{ABE}}, \hat{f})$*, and outputs* $\mathsf{sk}_f := (\hat{f}, \mathsf{sk}_{\hat{f}}^{\mathsf{ABE}})$ *as the secret key for* f*. Note that the description of* $\hat{f}$ *can be expressed as* $(f, \boldsymbol{y})$
- $\mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x})$*: Given a master public-key* mpk *and an attribute* $\boldsymbol{x} \in \{0, 1\}^*$ *as input, the algorithm first samples a random vector* $\boldsymbol{k} = (k_1, \ldots, k_n)^\top \in \mathbb{Z}_m^n$*, and then runs* $\mathsf{ABE.Enc}$ *mn times with attributes* $\boldsymbol{x}_{i,j} = (\boldsymbol{x}, i, j) \in \{0, 1\}^* \times [n] \times [m]$ *to set*

$$CT := \{ct_{i,j} \stackrel{\$}{\leftarrow} \mathsf{ABE.Enc}(\mathsf{mpk}, \boldsymbol{x}_{i,j}, k_i)\}_{(i,j) \in [n] \times [m]} \in \mathcal{CT}^{n \times m}, \text{ i.e.,}$$

$$CT := \begin{bmatrix} \mathsf{ABE.Enc}(\boldsymbol{x}_{1,1}, k_1) & \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,j}, k_1) & \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,m}, k_1) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{n,1}, k_n) & \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,j}, k_n) & \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,m}, k_n) \end{bmatrix}.$$

Finally, the algorithm outputs $(CT, \boldsymbol{k})$.

– $\mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x})$: *Given a master public-key* mpk *and an attribute* $\boldsymbol{x} \in \{0,1\}^*$ *as input, the algorithm first samples a random vector* $\boldsymbol{k} = (k_1, \ldots, k_n)^\top \in \mathbb{Z}_m^n$, *and then runs* $\mathsf{ABE.Enc}$ *mn times with attributes* $\boldsymbol{x}_{i,j} = (\boldsymbol{x}, i, j)$ *to set*

$$CT^* := \{ct^*_{i,j} \xleftarrow{\$} \mathsf{ABE.Enc}(\mathsf{mpk}, \boldsymbol{x}_{i,j}, k_i + j)\}_{(i,j)\in[n]\times[m]} \in \mathcal{CT}^{n\times m}, \text{ i.e.,}$$

$$CT^* := \begin{bmatrix} \mathsf{ABE.Enc}(\boldsymbol{x}_{1,1}, k_1{+}1) \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,j}, k_1{+}j) \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,m}, k_1{+}m) \\ \vdots \quad \ddots & \vdots \quad \ddots & \vdots \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{n,1}, k_n{+}1) \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,j}, k_n{+}j) \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,m}, k_n{+}m) \end{bmatrix},$$

where the addition $k_i + j$ *is performed over* $\mathbb{Z}_m$. *The algorithm outputs* CT^*.

– $\mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, CT)$: *Given a secret key* $\mathsf{sk}_f := (\boldsymbol{y}, \mathsf{sk}^{\mathsf{ABE}}_{\hat{f}})$ *and* $CT := \{ct_{i,j}\}_{(i,j)\in[n]\times[m]}$ *as input, the algorithm runs* $\mathsf{ABE.Dec}$ *to compute* $k_i = \mathsf{ABE.Dec}(\mathsf{sk}^{\mathsf{ABE}}_{\hat{f}}, ct_{i,y_i})$ *for all* $i \in [n]$, *and then outputs* $\boldsymbol{k} = (k_1, \ldots, k_n)^\top$, *if* $\hat{f}(\boldsymbol{x}, i, y_i) = f(\boldsymbol{x}) \wedge g_{\boldsymbol{y}}(i, y_i) = 1$ *for all* $i \in [n]$, *and* $\perp$ *otherwise.*

Intuitively, our attribute design (the class $\mathcal{G}$) allows the secret key to open one ciphertext per row while keeping the others secret. For the valid encapsulation, all ciphertexts in a row encrypts the same element, while for the invalid encapsulation, they encrypt different elements. As the secret key can only open one per row, an adversary cannot distinguish a valid from an invalid encapsulation, even given the secret key.

Our $\mathsf{AB\text{-}wHPS}$ secret key would be of length $|\hat{f}_{f,g_{\boldsymbol{y}}}| + s(\hat{f}_{f,g_{\boldsymbol{y}}}) = |\boldsymbol{y}| + |f| + s(\hat{f}_{f,g_{\boldsymbol{y}}}) = n \log m + |f| + s(\hat{f}_{f,g_{\boldsymbol{y}}})$, where $s(\cdot)$ is the key-size function (of the extra part, excluding the function description) of the underlying ABE. If the underlying ABE has succinct keys, i.e., $s(f) = o(|f|)$, then our $\mathsf{AB\text{-}wHPS}$ secret would have size $n \log m + |f| + s(\hat{f}_{f,g_{\boldsymbol{y}}}) = n \log m + |f| + o(n \log m + |f|)$. By setting sufficiently large n, m, we can achieve ABE with the optimal leakage rate, ref. Sect. 5.

Next we present the following theorem and its proof.

Theorem 3.12 ($\mathsf{AB\text{-}wHPS}$ from ABE) *Suppose* Π_{ABE} *is a secure* ABE *scheme with attribute space* $\bar{\mathcal{X}}_\kappa = \mathcal{X}_\kappa \times \mathcal{X}'_\kappa = \{0,1\}^* \times \{[n] \times [m]\}$ *for the function class* $\mathcal{F} \wedge_{\|} \mathcal{G}$, *where* $\mathcal{G}$ *is the class as in Definition 3.9 with parameters* m, n, *then the construction* $\Pi_{\mathsf{AB\text{-}wHPS}}$ *described above is an* $(n \log m, n \log m)$*-universal* $\mathsf{AB\text{-}wHPS}$ *with the attribute space* $\mathcal{X}_\kappa$ *and the encapsulated-key-space* $\mathcal{K} = \mathbb{Z}_m^n$, *for the function class* $\mathcal{F}$. *Furthermore,*

– *if the* ABE *is* X*-sel secure for* $\mathsf{X} \in \{sel, ada\}$, *then the* $\mathsf{AB\text{-}wHPS}$ *is* X *secure;*
– *if the key-size (of the extra part, excluding the function description) of the* ABE *scheme for policy function* f *is* $s(f)$, *then the key size of the* $\mathsf{AB\text{-}wHPS}$ *for* f *is* $n \log m + |f| + s(\hat{f}_{f,g_{\boldsymbol{y}}})$, *where* $s(\cdot)$ *is the key-size function (of the extra part, excluding the function description) of the underlying* ABE.

Proof The second part of the theorem follows directly by our construction from ABE to $\mathsf{AB\text{-}wHPS}$, especially by the relationship between policy functions of ABE and that of $\mathsf{AB\text{-}wHPS}$.

To prove the first part of the theorem, we need to prove the following three properties: correctness, smoothness and ciphertext indistinguishability.

Correctness Correctness of our $\mathsf{AB\text{-}wHPS}$ follows directly from the correctness of the underlying ABE.

Universality Given the master public key mpk and an invalid ciphertext $\mathsf{CT}^* = \mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x}) = \{\mathsf{ABE.Enc}(\boldsymbol{x}_{i,j}, k_i + j)\}_{i\in[n], j\in[m]}$, we have

$$\mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, \mathsf{CT}^*) = \boldsymbol{k} + \boldsymbol{y}$$

where $\mathsf{sk}_f := (\boldsymbol{y}, \mathsf{sk}_{\hat{f}_{f,g_{\boldsymbol{y}}}})$ for a randomly and independently chosen vector $\boldsymbol{y} = (y_1, \ldots, y_n)$, and $\boldsymbol{k}$ is the vector used to generate the invalid ciphertext. Clearly, the decryption function can be written as the permutation $\mathfrak{h}_{\boldsymbol{k}}(\boldsymbol{y}) = \boldsymbol{k} + \boldsymbol{y}$.

As this is an injective function of $\boldsymbol{y}$ (for any fixed $\boldsymbol{k}$), the min-entropy of $\boldsymbol{y}$ remains the same after applying this function, i.e.,

$$\begin{aligned} H_\infty(\mathsf{AB\text{-}wHPS.Decap}(\mathsf{CT}^*, \mathsf{sk}_f)|\mathsf{mpk}, \mathsf{CT}^*, \boldsymbol{x}) &= H_\infty((\boldsymbol{k} + \boldsymbol{y})|\mathsf{mpk}, \boldsymbol{x}, \mathsf{CT}^*) \\ &= H_\infty(\boldsymbol{y}|\mathsf{mpk}, \boldsymbol{x}, \mathsf{CT}^*). \end{aligned}$$

Moreover, we note that $\boldsymbol{y}$ is independent of $\mathsf{mpk}, \boldsymbol{x}, \mathsf{CT}^*$, so $H_\infty(\boldsymbol{y}|\mathsf{mpk}, \boldsymbol{x}, \mathsf{CT}^*) = n \log m$. As a result, the construction $\Pi_{\mathsf{AB\text{-}wHPS}}$ is (l, w)-universal, where $l = w = n \log m$.

Ciphertext indistinguishability We prove that the ciphertexts output by $\mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x}^*)$ and $\mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x}^*)$ are indistinguishable, given one secret "1-key" sk_f such that $f(\boldsymbol{x}^*) = 1$ and perhaps many "0-keys" $\mathsf{sk}_{f'}$ such that $f'(\boldsymbol{x}^*) = 0$, where $\boldsymbol{x}^*$ is the challenge attribute. We summarize the result in the lemma below. □

Lemma 3.13 (Ciphertext indistinguishability) *The construction of* AB-wHPS *satisfies selective (or adaptive) valid/invalid cipheretext indistinguishability as Definition 3.1, following from the sel-ada/sel-sel (or ada-ada/ada-sel) security of the underlying* ABE.

Proof To facilitate the proof presentation, we introduce an intermediate notion denoted as multi-ABE (with parameter t), where the adversary can send two challenge messages vectors $\boldsymbol{k}_0 = (k_{0,1}, \ldots, k_{0,t}) \in \mathbb{Z}_m^n$ and $\boldsymbol{k}_1 = (k_{1,1}, \ldots, k_{1,t}) \in \mathbb{Z}_m^n$, along with t different attributes $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$ as the challenge attributes. The adversary then receives a vector of challenge ciphertexts $\{c_i \leftarrow \mathsf{ABE.Enc}(\boldsymbol{x}_i, k_{b,i})\}_{i\in[t]}$ for a random bit b, and needs to decide a bit b'. Here the adversary is allowed to query sk_f as long as $f(\boldsymbol{x}_i) = 0$ for all $i \in [t]$, i.e., the key cannot open any component in the challenge ciphertexts. It is not hard to prove a reduction from the standard ABE to this multi-ABE via a hybrid argument, which only incurs a security loss t. □

Claim 3.14 *For any $t \in \mathbb{N}$, if there exists an adversary $\mathcal{A}$ that breaks the (partially) selective/adaptive security of multi-*ABE *with parameter t and advantage ε, then there exists a reduction $\mathcal{B}$ that breaks the same (partially) selective/adaptive security of* ABE *with advantage ε/t.*

Proof This follows from a standard hybrid argument. □

Next, we prove the valid/invalid ciphertext indistinguishability of AB-wHPS via a hybrid argument. We define the following hybrids, where we start from a valid ciphertext, and then switch row-by-row towards an invalid ciphertext. We prove that each two neighboring hybrids are indistinguishable via a reduction from multi-ABE (with parameter $m - 1$). The proof of this lemma follows directly from the indistinguishability of these hybrids.

Hybrid H_0: This hybrid is defined as the ciphertext indistinguishability experiment in Definition 3.1, where $\mathcal{A}$ is given a valid ciphertext

$$\mathsf{CT}_0 := \begin{bmatrix} \mathsf{ABE.Enc}(\boldsymbol{x}_{1,1}, k_1) & \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,m}, k_1) \\ \vdots & \ddots & \vdots \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{n,1}, k_n) & \ldots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,m}, k_n) \end{bmatrix} \in \mathcal{CT}^{n\times m}.$$

In this hybrid, it is clear that the ciphertext is generated as Encap.

Hybrid H_z: For any $1 \le z \le n-1$, H_z is almost same to H_{z-1}, except that $\mathcal{A}$ is given the following ciphertext

$$\mathsf{CT}_z := \begin{bmatrix} \mathsf{ABE.Enc}(\boldsymbol{x}_{1,1}, k_1+1) & \dots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,m}, k_1+m) \\ \vdots & \ddots & \vdots \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{z,1}, k_z+1) & \dots & \mathsf{ABE.Enc}(\boldsymbol{x}_{z,m}, k_z+m) \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{z+1,1}, k_{z+1}) & \dots & \mathsf{ABE.Enc}(\boldsymbol{x}_{z+1,m}, k_{z+1}) \\ \vdots & \ddots & \vdots \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{n,1}, k_n) & \dots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,m}, k_n) \end{bmatrix} \in \mathcal{CT}^{n\times m}.$$

In this hybrid, the first z rows are generated as Encap^* (that encrypts different keys), and the rest is as Encap (that encrypts the same key).

Hybrid H_n: This hybrid is almost same to H_{n-1}, except that $\mathcal{A}$ is given the following ciphertext

$$\mathsf{CT}_n := \begin{bmatrix} \mathsf{ABE.Enc}(\boldsymbol{x}_{1,1}, k_1+1) & \dots & \mathsf{ABE.Enc}(\boldsymbol{x}_{1,m}, k_1+m) \\ \vdots & \ddots & \vdots \\ \mathsf{ABE.Enc}(\boldsymbol{x}_{n,1}, k_n+1) & \dots & \mathsf{ABE.Enc}(\boldsymbol{x}_{n,m}, k_n+m) \end{bmatrix} \in \mathcal{CT}^{n\times m},$$

In this hybrid, it is clear that the ciphertext is generated as Encap^*.

Then, it suffices to prove the computational indistinguishability between H_z and H_{z+1} for $z \in [n-1]$

Claim 3.15 *Suppose the basic multi-*ABE *(with parameter* $m-1$*) is secure, then the above hybrids* H_z *and* H_{z+1} *are computational indistinguishability for any* $z \in [n-1]$.

Proof We prove this claim through establishing a reduction from the (partially) selective/adaptive security of multi-ABE to the corresponding indistinguishability between H_z and H_{z+1}. This means if there is an efficient adversary $\mathcal{D}$ who can distinguish H_z from H_{z+1} with advantage ε, then we can construct an efficient reduction $\mathcal{B}$ to break the corresponding multi-ABE with advantage ε. Here, we just describe the reduction in the case of $\mathsf{ada\text{-}sel}$ security (multi-ABE), and note that a similar argument can be carried to the $\mathsf{sel\text{-}ada/ada\text{-}ada/sel\text{-}sel}$ security in a straight-forward way.

In particular, let $\mathcal{A}$ be the adaptive adversary for the $\mathsf{AB\text{-}wHPS}$ with attribute space $\mathcal{X}_\kappa$ for the policy function class $\mathcal{F}$, and $\mathcal{D}$ be a distinguisher that distinguishes H_z from H_{z+1} with a non-negligible advantage for some $z \in [n-1]$. Now we describe the reduction $\mathcal{B}$ that breaks the $\mathsf{ada\text{-}sel}$ security of multi-ABE with attribute space $\mathcal{X}_\kappa \times \{[n] \times [m]\}$ for the policy function class $\mathcal{F} \wedge_{\|} \mathcal{G}$, when interacting with the challenger $\mathcal{C}$.

Setup $\mathcal{B}$ simulates either the hybrid H_z or H_{z+1} by running $\mathcal{A}$ in the following way.

1. With respect to the $\mathsf{ada\text{-}sel}$ security of multi-ABE, $\mathcal{B}$ selectively chooses $(m-1)$ attributes $(z+1,2), \dots, (z+1,m) \in [n] \times [m]$, and then sends them to $\mathcal{C}$ before getting mpk, where $(z+1,2), \dots, (z+1,m)$ are essentially the second part of challenge attributes for multi-ABE;
2. $\mathcal{B}$ gets a master public-key mpk from the challenger $\mathcal{C}$ for the multi-ABE.
3. Then $\mathcal{B}$ forwards this mpk to the adversary $\mathcal{A}$ for the $\mathsf{AB\text{-}wHPS}$.
4. At the same time, $\mathcal{B}$ initializes a table $T = \emptyset$.

Test Stage 1 $\mathcal{B}$ answers the secret key queries of $\mathcal{A}$ in the following way.

1. $\mathcal{A}$ sends a function $f \in \mathcal{F}$ to $\mathcal{B}$ for a secret key query.
2. $\mathcal{B}$ first checks whether there exists an item containing this f in the table T.
   - If yes, $\mathcal{B}$ returns the corresponding secret key sk_f in T to $\mathcal{A}$.
   - Otherwise, $\mathcal{B}$ goes to the next step 3.
3. $\mathcal{B}$ chooses a random vector $\boldsymbol{y} \xleftarrow{\$} [m]^n$ such that $g_{\boldsymbol{y}}(z+1, j) = 0$ for all $2 \le j \le m$, and sets $\hat{f} := \hat{f}_{f,g_{\boldsymbol{y}}} \in \mathcal{F} \wedge_{\|} \mathcal{G}$.
4. Then $\mathcal{B}$ sends this $\hat{f}$ to $\mathcal{C}$ as a secret key query for multi-ABE, and then gets $\mathsf{sk}_{\hat{f}}^{\mathsf{ABE}}$ as its response.
5. Finally, $\mathcal{B}$ sends $\mathsf{sk}_f := (\boldsymbol{y}, \mathsf{sk}_{\hat{f}}^{\mathsf{ABE}})$ as the secret key for f to $\mathcal{A}$, and stores the tuple $(f, \boldsymbol{y}, \mathsf{sk}_{\hat{f}}^{\mathsf{ABE}}))$ as an item in the table T.

Challenge stage

$\mathcal{B}$ simulates the challenge ciphertext to $\mathcal{A}$ as follows.

1. With respect to the adaptive security of AB-wHPS, $\mathcal{A}$ adaptively selects an attribute $\boldsymbol{x}^* \in \mathcal{X}_\kappa$ and sends it to $\mathcal{B}$.
2. $\mathcal{B}$ chooses a random values $k \xleftarrow{\$} \mathbb{Z}_m$, and uses k to set two sequences of messages
$$\boldsymbol{k}_0 = (k_{0,2}, \ldots, k_{0,m})^\top = (k, \ldots, k)^\top \in \mathbb{Z}_m^{m-1}$$
and
$$\begin{aligned}\boldsymbol{k}_1 &= (k_{1,2}, \ldots, k_{1,m})^\top \\ &= (k+1, \ldots, k+m-1)^\top \in \mathbb{Z}_m^{m-1}.\end{aligned}$$
3. Then $\mathcal{B}$ sends $(\boldsymbol{k}_0, \boldsymbol{k}_1)$ and the attribute $\boldsymbol{x}^*$ as the challenge query of multi-ABE, where $\boldsymbol{x}^*$ composes of the first part of challenge attributes for multi-ABE.
4. As a result, $\mathcal{B}$ obtains $(m-1)$ ciphertexts
$$\left\{\mathsf{ct}_{z+1,j}^* \xleftarrow{\$} \mathsf{ABE.Enc}(\boldsymbol{x}_{z+1,j}^*, k_{b,j})\right\}_{2 \le j \le m}$$
for a random $b \in \{0, 1\}$ chosen by the multi-ABE challenger $\mathcal{C}$, where $\{\boldsymbol{x}_{z+1,j}^* = (\boldsymbol{x}^*, z+1, j)\}_{2 \le j \le m}$.
5. Furthermore, $\mathcal{B}$ chooses $(n-1)$ random values $v_1, \ldots, v_i, v_{i+2}, \ldots, v_n \xleftarrow{\$} \mathbb{Z}_m$.
6. $\mathcal{B}$ sets $\boldsymbol{x}_{i,j}^* = (\boldsymbol{x}^*, i, j)$, and then calculates
$$\left\{\mathsf{ct}_{i,j}^* \xleftarrow{\$} \mathsf{ABE.Enc}(\boldsymbol{x}_{i,j}^*, v_i + j)\right\}_{i \in [z], j \in [m]},$$
$$\left\{\mathsf{ct}_{i,j}^* \xleftarrow{\$} \mathsf{ABE.Enc}(\boldsymbol{x}_{i,j}^*, v_i)\right\}_{i \in [n] \setminus [z+1], j \in [m]}$$
and
$$\mathsf{ct}_{z+1,w}^* \xleftarrow{\$} \mathsf{ABE.Enc}(\boldsymbol{x}_{z+1,1}^*, k).$$
7. $\mathcal{B}$ collects all ciphetexts $\mathsf{ct}_{i,j}^*$ for $i \in [n]$, $j \in [m]$ together to construct a $n \times m$ matrix CT^* according to the indices of these ciphertexts.
8. Finally, $\mathcal{B}$ sends this matrix CT^* as the challenge encapsulation ciphertext to $\mathcal{A}$.

Test Stage 2 $\mathcal{B}$ answers the secret key queries of $\mathcal{A}$ as in Test Stage 1, but with a restriction that there is at most one function $f \in \mathcal{F}$ such that $f(x^*) = 1$ can been queried in Test Stage 1 and 2.

Output $\mathcal{B}$ simulates the output of the experiment according to the response of $\mathcal{A}$, and thus obtain a view H, which is either H_z or H_{z+1} as we will prove below. Finally, $\mathcal{B}$ outputs $\mathcal{D}(\mathsf{H})$.

Next, we analyze the advantage of $\mathcal{B}$. We observe that $\mathcal{B}$ perfectly simulates one of the two hybrids: if the challenge ciphertext from $\mathcal{C}$ encrypts $\boldsymbol{k}_0$, then the AB-wHPS challenge ciphertext CT^* is generated according to H_z, and otherwise H_{z+1}. Thus, the advantage of $\mathcal{B}$ is the same as that of $\mathcal{D}$ in distinguishing H_z from H_{z+1}, i.e., a non-negligible advantage ε. Thus, $\mathcal{B}$ breaks the multi-ABE with advantage ε, which reaches a contradiction. This completes the proof of this claim. □

Lemma 3.13 follows directly from Claim 3.15 by a standard hybrid argument. □
In summary, we complete the proof of the first part of theorem. □

4 Instantiations of AB-wHPS from lattices

In this section, we present concrete instantiations of AB-wHPS from lattices. In order to do this, according to the generic construction in Sect. 3.2, we just need to present the underlying ABE with the corresponding properties. Notice that as ABE in [10] supports general circuits as policy function class, it implicitly implies sel-sel secure ABE for $\mathcal{F} \wedge_{\|} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are general boolean circuits.

Besides, we instantiate two partial-adaptively secure ABE schemes as needed in Sect. 3.2 from LWE with a polynomial modulus. The first construction is with respect to the function family $\mathcal{I} \wedge_{\|} \mathcal{G}$, where $\mathcal{I}$ is the equation test function family for which a function $\mathsf{id} \in \mathcal{I}$ satisfies $f_{\mathsf{id}}(\boldsymbol{x}) = 1$ if and only if $\mathsf{id} = (b_1, \ldots, b_\ell) = \boldsymbol{x}$ and 0 otherwise, and $\mathcal{G}$ is a general circuit family. The second construction is with respect to the function family $(t\text{-}\mathsf{CNF}^*) \wedge_{\|} \mathcal{G}$.

Particularly, our first construction combines the adaptively secure IBE scheme proposed by Agrawal et al. [3] and the selectively secure ABE proposed by Boneh et al. [10] in a natural way, and achieves the ada-sel security. The second construction combines the recent ABE scheme by Tsabary [10, 40], and obtains the ada-sel security. We present our first construction in Sect. 4.1.1, and the second in Sect. 4.2.

4.1 ada-sel secure ABE based on LWE

4.1.1 Construction of ABE for $\mathcal{I} \wedge_{\|} \mathcal{G}$ from lattices

For convenience, we denote $\mathcal{I} \wedge_{\|} \mathcal{G}$ as $\mathcal{F}_1$ for short.

ABE.Setup$_{\mathcal{F}_1}(\mathbf{1}^\kappa)$: The setup algorithm takes as input a security parameter κ, and then does the following:

1. Sample a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ along with a trapdoor basis $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m \times m}$ of lattice $\Lambda_q^\perp(\mathbf{A})$ by running TrapGen.
2. Select $\ell_1 + 1$ uniformly random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_{\ell_1}, \mathbf{B} \in \mathbb{Z}_q^{n \times m}$.
3. Select ℓ_2 uniformly random matrices $\mathbf{C}_1, \ldots, \mathbf{C}_{\ell_2} \in \mathbb{Z}_q^{n \times m}$.
4. Select a random matrix $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n \times z}$.

5. Output the public parameters

$$\mathsf{mpk} = (\mathbf{A}, \{\mathbf{A}_i\}_{i\in[\ell_1]}, \{\mathbf{C}_i\}_{i\in[\ell_2]}, \mathbf{B}, \mathbf{U})$$

and the master secret key $\mathsf{msk} = (\mathbf{T}_{\mathbf{A}})$.

ABE.KeyGen$_{\mathcal{F}_1}$**(mpk, mpk,** $f_{\mathsf{id}} \wedge_{\|} g$**)**: The key generation algorithm takes as input mpk, msk, an equation test function for id with binary representation $(b_1, b_2, \ldots, b_{\ell_1}) \in \{0,1\}^{\ell_1}$ and a policy function $g \in \mathcal{G}$ with depth d, and then does the following:

1. Compute $\mathbf{A}_{\mathsf{id}} = \mathbf{B} + \sum_{i=1}^{\ell_1}(b_i\mathbf{A}_i) \in \mathbb{Z}_q^{n\times m}$.
2. Define function $\bar{g}(\cdot) = 1 - g(\cdot)$, and compute

$$\mathbf{H}_g = \mathsf{Eval}_{\mathsf{pk}}(\bar{g}, \mathbf{C}_1, \ldots, \mathbf{C}_{\ell_2}) \in \mathbb{Z}_q^{n\times m}.$$

3. Let $\mathbf{F}_{f_{\mathsf{id}}\wedge_{\|}g} = (\mathbf{A}|\mathbf{A}'_{f_{\mathsf{id}}\wedge_{\|}g}) = (\mathbf{A}|\mathbf{A}_{\mathsf{id}}|\mathbf{H}_g) \in \mathbb{Z}_q^{n\times 3m}$.
4. Sample $\mathbf{D} \in \mathbb{Z}^{3m\times z}$ as $\mathbf{D} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{A}'_{f_{\mathsf{id}}\wedge_{\|}g}, \mathbf{U}, \tau)$.
5. Output $\mathsf{sk}_{f_{\mathsf{id}}\wedge_{\|}g} := \mathbf{D}$, where $\mathbf{F}_{f_{\mathsf{id}}\wedge_{\|}g} \cdot \mathbf{D} = \mathbf{U} \bmod q$.

ABE.Enc$_{\mathcal{F}_1}$**(mpk,** $\boldsymbol{x_1}, \boldsymbol{x_2}$**),** $\boldsymbol{\mu}$**)**: In order to encrypt a message $\boldsymbol{\mu} \in \{0,1\}^z$ with respect to attribute $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ where $\boldsymbol{x}_1 = (x_{11}, \ldots, x_{1\ell_1}) \in \{0,1\}^{\ell_1}$ and $\boldsymbol{x}_2 = (x_{21}, \ldots, x_{2\ell_2}) \in \mathbb{Z}_q^{\ell_2}$, the encryption algorithm first chooses a random vector $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$ and two error vectors $\boldsymbol{e}_0 \leftarrow \chi^m$, $\boldsymbol{e}_1 \leftarrow \chi^z$ where χ is a B bounded discrete Gaussian distribution, and then does the following:

1. Compute $\mathbf{A}_{\boldsymbol{x}_1} = \mathbf{B} + \sum_{i=1}^{\ell_1}(x_{1i}\mathbf{A}_i) \in \mathbb{Z}_q^{n\times m}$.
2. Choose ℓ_1 uniformly random matrices $\mathbf{R}_i \leftarrow \{-1,1\}^{m\times m}$ for $i \in [\ell_1]$, and compute $\mathbf{R}_{\boldsymbol{x}_1} = \sum_{i=1}^{\ell_1}(x_{1i}\mathbf{R}_i)$.
3. Set $\boldsymbol{e}_2 = \mathbf{R}_{\boldsymbol{x}_1}^{\top} \cdot \boldsymbol{e}_0 \in \mathbb{Z}_q^m$.
4. Set $\mathbf{H}_{\boldsymbol{x}_2} = (x_{21}\mathbf{G} + \mathbf{C}_1| \cdots |x_{2\ell_2}\mathbf{G} + \mathbf{C}_{\ell_2}) \in \mathbb{Z}_q^{n\times m\ell_2}$.
5. Choose ℓ_2 uniformly random matrices $\mathbf{R}'_j \leftarrow \{-1,1\}^{m\times m}$ for $j \in [\ell_2]$, and set $\boldsymbol{e}_3 = (\mathbf{R}'_1| \cdots |\mathbf{R}'_{\ell_2})^{\top} \cdot \boldsymbol{e}_0 \in \mathbb{Z}_q^{m\ell_2}$.
6. Set $\mathbf{F}_{\boldsymbol{x}_1,\boldsymbol{x}_2} = (\mathbf{A}|\mathbf{A}'_{\boldsymbol{x}_1,\boldsymbol{x}_2}) = (\mathbf{A}|\mathbf{A}_{\boldsymbol{x}_1}|\mathbf{H}_{\boldsymbol{x}_2}) \in \mathbb{Z}_q^{n\times(2+\ell_2)m}$.
7. Output $\boldsymbol{c} = (\mathbf{F}_{\boldsymbol{x}_1,\boldsymbol{x}_2}^{\top} \cdot \boldsymbol{s} + (\boldsymbol{e}_0^{\top}, \boldsymbol{e}_2^{\top}, \boldsymbol{e}_3^{\top})^{\top}, \mathbf{U}^{\top} \cdot \boldsymbol{s} + \boldsymbol{e}_1 + \lfloor q/2 \rceil \boldsymbol{\mu}) \in \mathbb{Z}_q^{(2+\ell_2)m+z}$.

ABE.Dec$_{\mathcal{F}_1}$**(mpk,** $\mathsf{sk}_{f_{\mathsf{id}}\wedge_{\|}g}$**,** $(\boldsymbol{x}, \boldsymbol{c})$**)**: The decryption algorithm uses the key $\mathsf{sk}_{f_{\mathsf{id}}\wedge_{\|}g} := \mathbf{D}$ to decrypt $\boldsymbol{c}$ with attribute $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2)$. If $f_{\mathsf{id}}(\boldsymbol{x}_1) \wedge g(\boldsymbol{x}_2) \neq 1$, output $\perp$. Otherwise, let the ciphertext $\boldsymbol{c} = (\boldsymbol{c}_{in,1}, \boldsymbol{c}_{in,2}, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{\ell_2}, \boldsymbol{c}_{out}) \in \mathbb{Z}_q^{(2+\ell_2)m+z}$, compute $\boldsymbol{c}_g = \mathsf{Eval}_{ct}(\bar{g}, \{(x_i, \mathbf{C}_i, \boldsymbol{c}_i)\}_{i=1}^{\ell_2}) \in \mathbb{Z}_q^m$, where $\boldsymbol{c}_{in,1}, \boldsymbol{c}_{in,2} \in \mathbb{Z}_q^m$, $\boldsymbol{c}_{out} \in \mathbb{Z}_q^z$ and $\boldsymbol{c}_i \in \mathbb{Z}_q^m$ for $1 \le i \le \ell_2$.

Let $\boldsymbol{c}'_g = (\boldsymbol{c}_{in,1}, \boldsymbol{c}_{in,2}, \boldsymbol{c}_g) \in \mathbb{Z}_q^{3m}$ and output $\mathsf{Round}(\boldsymbol{c}_{out} - \mathbf{D}^{\top} \cdot \boldsymbol{c}'_g) \in \{0,1\}^m$.

Correctness. The correctness of the scheme follows from our choice of parameters. Specifically, to show correctness first note that when $f_{\mathsf{id}}(\boldsymbol{x}_1) \wedge g(\boldsymbol{x}_2) = 1$ we know $\boldsymbol{c}_{in,2} = \mathbf{A}_{\mathsf{id}}^{\top} \cdot \boldsymbol{s} + \boldsymbol{e}_2$, $\boldsymbol{c}_g = \mathbf{H}_g^{\top} \cdot \boldsymbol{s} + \boldsymbol{e}_g$, then we have during decryption,

$$\begin{aligned}
\boldsymbol{\mu}' &= \mathsf{Round}(\boldsymbol{c}_{out} - \mathbf{D}^{\top} \cdot \boldsymbol{c}'_g) \\
&= \mathsf{Round}(\boldsymbol{c}_{out} - \mathbf{D}^{\top} \cdot ((\mathbf{A}|\mathbf{A}_{\mathsf{id}}|\mathbf{H}_g)^{\top} \cdot \boldsymbol{s} + (\boldsymbol{e}_0, \boldsymbol{e}_2, \boldsymbol{e}_g))) \\
&= \mathsf{Round}(\mathbf{U}^{\top} \cdot \boldsymbol{s} + \boldsymbol{e}_1 + \lfloor q/2 \rceil \boldsymbol{\mu} - \mathbf{U}^{\top} \cdot \boldsymbol{s} - \mathbf{D}^{\top} \cdot (\boldsymbol{e}_0, \boldsymbol{e}_2, \boldsymbol{e}_g)) \\
&= \mathsf{Round}(\lfloor q/2 \rceil \boldsymbol{\mu} + \boldsymbol{e}_1 - \mathbf{D}^{\top} \cdot (\boldsymbol{e}_0, \boldsymbol{e}_2, \boldsymbol{e}_g)) \\
&= \boldsymbol{\mu}
\end{aligned}$$

Table 2 Parameter setting of ada-sel secure ABE for $\mathcal{I} \wedge_{\|} \mathcal{G}$

Parameters	Description	Setting
κ	Security parameter	
z	Message length	$O(\log \kappa)$
n	PK-lattice row dimension	κ
m	PK-lattice column dimension	$n^{1+\epsilon}$
q	Modulus	$n^5 m^4$
d	Depth of $g \in \mathcal{G}$	$O(\log \kappa)$
τ	SampleLeft and SampleRight parameter	$n^2 m^2$
B	Bound of errors	κ
ℓ_1	Identity length	n
ℓ_2	Attribute length	n

This completes the proof of correctness.

4.1.2 Parameter setting for our construction

Given an arbitrarily constant ϵ, we set the system parameters according to the Table 2 below. These values are chosen in order to satisfy the following constraints:

- To ensure correctness, we require $\|\boldsymbol{e}_1 - \mathbf{D}^\top \cdot (\boldsymbol{e}_0, \boldsymbol{e}_2, \boldsymbol{e}_g)\|_\infty \leq q/4$; here we bound the dominating term:

$$\|\mathbf{D}^\top \cdot \boldsymbol{e}_g\|_\infty \leq \tau\sqrt{3m} \cdot 4^d m^{3/2} B \leq q/4.$$

- For SampleLeft, we know $\|\widetilde{\mathbf{T}_\mathbf{A}}\| = O(\sqrt{n \log q})$, so require that the sampling width τ satisfies

$$\tau \geq O(\sqrt{n \log q}) \cdot \omega(\sqrt{\log 3m}).$$

- For SampleRight. we know $\|\widetilde{\mathbf{T}_\mathbf{G}}\| \leq \sqrt{5}$ and that

$$\tau \geq \sqrt{5} \cdot 4^d m^{3/2} \cdot \omega(\sqrt{\log m}) \geq \|\widetilde{\mathbf{T}_\mathbf{G}}\| \cdot s_{\mathbf{R}_g} \cdot \omega(\sqrt{\log m}).$$

- To apply Regev's reduction, we need $B \geq \sqrt{n}\omega(\log n)$.
- To apply the Leftover Hash Lemma, we need $m > (n+1)\log q + \omega(\log n)$.

4.1.3 Secret key size

We give a simple analysis of the secret key size of our $\mathsf{ABE}_{\mathcal{F}_1}$ construction. By Lemma 2.2, we know that

$$\Pr[\mathbf{D} \leftarrow D_{\Lambda_q^\perp(\mathbf{F}_{f_{\mathsf{id}} \wedge_{\|} g}), \tau} : \|\mathbf{D}\| > \tau\sqrt{3m}] \leq \mathsf{negl}(n).$$

By our setting of parameters above, the size of the secret key of our ABE scheme for $\mathcal{F}_1$ is bounded by $O(\kappa^{1+\epsilon} \log^2 \kappa)$.

4.1.4 Security proof of $\mathsf{ABE}_{\mathcal{F}_1}$

Below, we prove the security of $\mathsf{ABE}_{\mathcal{F}_1}$ in a formal way.

Theorem 4.1 *For parameter setting in Table 2,* $\mathsf{ABE}_{\mathcal{F}_1}$ *scheme above is ada-sel secure as defined in Definition 3.5 and Remark 3.6, assuming the* $\mathsf{LWE}_{n,q,\chi}$ *assumption holds.*

Proof We prove the security of $\mathsf{ABE}_{\mathcal{F}_1}$ construction by a sequence of hybrids, where the first hybrid is identical to the original security experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{ada\text{-}sel}}(1^\kappa)$ as in Definition 3.5. We show that if a PPT adversary $\mathcal{A}$ that makes at most $|Q|$ secret key queries, can break the $\mathsf{ABE}_{\mathcal{F}_1}$ scheme described above with non-negligible advantage ε (i.e. success probability $1/2+\varepsilon$), then there exists a reduction that can break the LWE assumption with advantage $\mathsf{poly}(\varepsilon) - \mathsf{negl}(\varepsilon)$. Given such an adversary $\mathcal{A}$, we consider the following hybrids. In **Hybrid** H_i we let W_i denote the event that the adversary correctly guessed the challenge bit, namely that $b = b'$ at the end of the game. The adversary's advantage in H_i is $|\mathsf{Pr}[W_i] - \frac{1}{2}|$.

The Sequence of Hybrids ($\mathbf{H_0}$, $\mathbf{H_1}$, $\mathbf{H_2}$, $\mathbf{H}_3$, $\mathbf{H_4}$)

Hybrid H_0: This is the original security experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{ada\text{-}sel}}(1^\kappa)$ from Definition 3.5 between the adversary $\mathcal{A}$ and the challenger.

Hybrid H_1: In hybrid H_1, we slightly change the way that the challenger generates the matrices $\mathbf{A}_i$ for $i \in [\ell_1]$ and the matrices $\mathbf{C}_j$ for $j \in [\ell_2]$ in the public parameters. We let $\mathbf{R}_i \in \{-1,1\}^{m\times m}$ for $i \in [\ell_1]$ and $\mathbf{R}'_j \in \{-1,1\}^{m\times m}$ for $j \in [\ell_2]$ denote the $\ell_1+\ell_2$ ephemeral random matrices generated for the creation of ct^*. The hybrid H_1 challenger chooses ℓ_1 random elements $h_i \in \mathbf{GF}(q^t)$. Next it generates matrices $\mathbf{A}$ and $\mathbf{B}$ as in H_0 and constructs the matrices $\mathbf{A}_i$ for $i \in [\ell_1]$ as

$$\mathbf{A}_i = \mathbf{A}\cdot\mathbf{R}_i + (G(h_i)\otimes \mathbf{I}_{n/t})\mathbf{G},$$

where G is the ring isomorphic map described in Sect. 2.5, and constructs $\mathbf{C}_j$ for $j \in [\ell_2]$ as

$$\mathbf{C}_j = \mathbf{A}\cdot\mathbf{R}'_j - x^*_{2j}\mathbf{G},$$

where $\boldsymbol{x}^*_2 = (x^*_{21},\ldots,x^*_{2\ell_2})^\top \in \{0,1\}^{\ell_2}$ is the challenge attribute.

We show that H_0 and H_1 are statistically indistinguishable. Observe that in H_1, the matrices $\mathbf{R}_i$ for $i \in [\ell_2]$ are used only in the construction of the matrices $\mathbf{A}_i$ and in the construction of the challenge ciphertext where $\boldsymbol{e}_2 = (\mathbf{R}_{\boldsymbol{x}^*_1})^\top\cdot\boldsymbol{e}_0 \in \mathbb{Z}_q^m$, where $\mathbf{R}_{\boldsymbol{x}^*_1} = \sum_{i=1}^{\ell_1} x^*_{1i}\mathbf{R}_i$. Let $\tilde{\mathbf{R}} = (\mathbf{R}_1|\ldots|\mathbf{R}_{\ell_1}|\mathbf{R}'_1|\ldots|\mathbf{R}'_{\ell_2}) \in \mathbb{Z}_q^{m\times(\ell_1+\ell_2)m}$ then by Lemma 2.6, the distributions

$$\left(\mathbf{A}, \mathbf{A}\cdot\tilde{\mathbf{R}}, (\tilde{\mathbf{R}})^\top\cdot\boldsymbol{e}_0\right) \overset{s}{\approx} \left(\mathbf{A}, (\mathbf{A}'_1|\ldots|\mathbf{A}'_{\ell_1+\ell_2}), (\tilde{\mathbf{R}})^\top\cdot\boldsymbol{e}_0\right)$$

are statistically close, where $\mathbf{A}'_i$ for $i \in [\ell_1+\ell_2]$ are uniform independent matrices in $\mathbb{Z}_q^{n\times m}$. It follows that with $\boldsymbol{e}_2 = (\mathbf{R}_{\boldsymbol{x}^*_1})^\top\cdot\boldsymbol{e}_0$ and $\boldsymbol{e}_3 = (\mathbf{R}'_1|\cdots|\mathbf{R}'_{\ell_2})^\top\cdot\boldsymbol{e}_0$ the distributions

$$\left(\mathbf{A}, \mathbf{A}\mathbf{R}_1,\ldots,\mathbf{A}\mathbf{R}_{\ell_1}, \mathbf{A}\mathbf{R}'_1,\ldots,\mathbf{A}\mathbf{R}'_{\ell_2}, \boldsymbol{e}_2, \boldsymbol{e}_3\right) \overset{s}{\approx} \left(\mathbf{A}, \mathbf{A}'_1,\ldots,\mathbf{A}'_{\ell_1+\ell_2}, \boldsymbol{e}_2, \boldsymbol{e}_3\right).$$

Therefore, in the adversary's view, the matrices $\mathbf{A}\mathbf{R}_i, \mathbf{A}\mathbf{R}'_j$ are statistically close to uniform and independent of $\boldsymbol{e}_2, \boldsymbol{e}_3$. Hence, matrices $\mathbf{A}_i$ and $\mathbf{C}_j$ as defined as above are close to uniform, which means that those matrices are random independent matrices in the attacker's view, as in H_0. This shows that $|\mathsf{Pr}[W_0] - \mathsf{Pr}[W_1]| = \mathsf{negl}(\kappa)$.

Hybrid H_2: Hybrid H_2 is identical to Hybrid H_1 except that we add an abort event that is independent of the adversary's view. The H_2 challenger behaves as follows:

– The setup phase is identical to H_1 except that the challenger also chooses a random hash function $H \in \mathcal{H}_{\mathsf{pind}}$ and keeps it to itself.
– The challenger responds to identity-policy queries and issues the challenge ciphertext exactly as in H_1 (using a random bit $b \in \{0, 1\}$ to select the type of challenge). Let $\big((f_{\mathsf{id}_1} \wedge_{\|} g_1), \ldots, (f_{\mathsf{id}_t} \wedge_{\|} g_t)\big)$ be the identity-policy pairs where the attacker queries and let $\boldsymbol{x}_1^*$ be the challenge identity and $\boldsymbol{x}_2^*$ be the challenge attribute. By definition, the two events that $\boldsymbol{x}_1^* \in \{\mathsf{id}_1, \ldots, \mathsf{id}_t\}$ and $g_i(\boldsymbol{x}_2^*) = 1$ for $i \in [t]$ can not happen at the same time.
– In the final guess phase, the attacker outputs its guess $b' \in \{0, 1\}$ for b. The challenger now does the abort check: $H(\boldsymbol{x}_1^*) = 0$ and $H(\mathsf{id}_i) \neq 0$ for all $\mathsf{id}_i \in \{\mathsf{id}_i\}_{i\in[t]} \backslash \{\boldsymbol{x}_1^*\}$. If the condition does not hold, the challenger overwrites b' with a freshly random bit in $\{0, 1\}$, and we say the challenge aborts the game.

Note that the adversary never sees the random hash function, and has no idea if an abort event took place. While it is convenient to describe the abort action at the end of the game, nothing would change if the challenger aborted the game as soon as the abort condition becomes true.
The only difference between hybrids H_0 and H_1 is the abort event. We argue that the adversary still has non-negligible advantage in H_1 even though the abort event can happen. More formally, we will use Lemma 28 in the full version of the work [3], which is described as follows.

□

Lemma 4.2 *Let I be a $Q + 1$ tuple $(\boldsymbol{x}_1^*, \mathsf{id}_1, \ldots, \mathsf{id}_{|Q|})$ consisting of the challenge attribute $\boldsymbol{x}_1^*$ along with the queried ID's, and let $\varepsilon(I)$ define the probability that an abort does not happen in hybrid H_i. For $i = 1, 2$, we set W_i be the event that $b = b'$ at the end of hybrid H_i. Assuming $\varepsilon(I) \in [\varepsilon_{min}, \varepsilon_{max}]$, then we have*

$$\left|\mathsf{Pr}[W_2] - \frac{1}{2}\right| \geq \varepsilon_{min} \left|\mathsf{Pr}[W_1] - \frac{1}{2}\right| - \frac{1}{2}(\varepsilon_{max} - \varepsilon_{min}).$$

The lemma was analyzed by Bellare and Ristenpart [9], and further elaborated in the work [3]. As our overall proof just uses this lemma in a "black-box way", we do not include its proof for simplicity of presentation.

Hybrid H_3: We now change how $\mathbf{A}$ and $\mathbf{B}$ in H_2 are chosen. In H_3 we generate $\mathbf{A}$ as a random matrix in $\mathbb{Z}_q^{n\times m}$, but generate $\mathbf{B}$ by sampling a random matrix $\mathbf{R} \in \{-1, 1\}^{m\times m}$ and computing $\mathbf{B} = \mathbf{A} \cdot \mathbf{R} + \mathbf{G} \in \mathbb{Z}_q^{n\times m}$. The construction of $\mathbf{A}_i$ for $i = 1, \ldots, \ell_1$ and $\mathbf{C}_j$ for $j = 1, \ldots, \ell_2$ remains as in H_2, namely, $\mathbf{A}_i = \mathbf{A} \cdot \mathbf{R}_i + (G(h_i) \otimes \mathbf{I}_{n/t})\mathbf{G}$. To respond to a private key query for $\mathsf{id} = (b_1, \ldots, b_{\ell_1}) \in \{0, 1\}^{\ell_1}$ and a policy function $g \in \mathcal{G}$ with depth d, the challenger needs to output a small matrix $\mathbf{D} \in \Lambda_q^{\mathbf{U}}(\mathbf{F}_{f_{\mathsf{id}} \wedge_{\|} g})$, where

$$\mathbf{F}_{f_{\mathsf{id}} \wedge_{\|} g} = \left(\mathbf{A}|\mathbf{B} + \sum_{i=1}^{\ell_1}(b_i\mathbf{A}_i)|\mathbf{H}_g\right) = (\mathbf{A}|\mathbf{A} \cdot \mathbf{R}_{\mathsf{id}} + H(\mathsf{id})\mathbf{G}|\mathbf{A} \cdot \mathbf{R}_g - (1 - g(\boldsymbol{x_2}^*))\mathbf{G})$$

with

$$\mathbf{R}_{\mathsf{id}} = \mathbf{R} + \sum_{i=1}^{\ell_1}(b_i\mathbf{R}_i) \text{ and } \mathbf{R}_g = \mathsf{Eval}(\bar{g}, \mathbf{A}, \mathbf{R}_1, \ldots, \mathbf{R}_{\ell_2}, \boldsymbol{x}_2^*)$$

$$\text{and} \quad H(\mathsf{id}) = \mathbf{I}_n + \sum_{i=1}^{\ell_1} b_i (G(h_i) \otimes \mathbf{I}_{n/t}).$$

Note that H is the hash function in $\mathcal{H}_{\mathsf{pind}}$ defined by $(h_1, \ldots, h_{\ell_1})$ as in Sect. 2.5.
The challenger now does the following:

1. Construct $H(\mathsf{id})$ and $\mathbf{R}_{\mathsf{id}}$ as in above. If $H(\mathsf{id}) = 0$ and $g(\boldsymbol{x}_2^*) = 1$ abort the game and pretend that the adversary outputs a random bit b' in $\{0, 1\}$, as in H_2.
2. Set $\mathbf{D} \leftarrow \mathsf{SampleRight}(\mathbf{A}, H(\mathsf{id}), \mathbf{R}_{\mathsf{id}}, \mathbf{T}_{\mathbf{G}}, \mathbf{U}, \sigma, \mathbf{R}_g) \in \mathbb{Z}^{3m \times z}$.
3. Send $\mathsf{sk}_{f_{\mathsf{id}} \wedge \| g} = \mathbf{D}$ to $\mathcal{A}$.

H_3 is otherwise the same as H_2. In particular, in the challenge phase the challenger checks if the challenge attribute $(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*) \in \{0, 1\}^{\ell_1 + \ell_2}$ satisfies $H(\boldsymbol{x}_1^*) = 0$ and $f(\boldsymbol{x}_2^*) = 1$. If not, the challenger aborts the game (and pretends that the adversary output a random bit b' in $\{0, 1\}$), as in H_2. Similarly, in H_3 the challenger implements an abort check in the guess phase.
Since H_2 and H_3 are statistically indistinguishable in the attacker's view (the public parameters, responses to private key queries, the challenge ciphertext, and abort conditions) the adversary's advantage in H_3 is statistically indistinguishable to its advantage in H_2, namely

$$|\mathsf{Pr}[W_3] - \mathsf{Pr}[W_2]| = \mathsf{negl}(\kappa).$$

Hybrid H_4: Hybrid H_4 is identical to H_3 except that the challenge ciphertext ct is always chosen as a random independent element in $\mathbb{Z}_q^{(2+\ell_2)m+z}$. Since the challenge ciphertext is always a fresh random element in the ciphertext space, $\mathcal{A}$'s advantage in this hybrid is zero.

It remains to show that H_3 and H_4 are computationally indistinguishable, which we do by giving a reduction from the LWE problem. If an abort event happens then the games are clearly indistinguishable. Therefore, it suffice to focus on sequences of queries that do not cause an abort. We have the following lemma:

Lemma 4.3 *Assuming the hardness of* LWE *assumption, hybrid* H_3 *and* H_4 *are computationally indistinguishable.*

Proof Suppose there exists an adversary who has non-negligible advantage in distinguishing hybrid H_3 and H_4, then we can construct a reduction B that breaks the LWE assumption using the adversary $\mathcal{A}$. Recall in Definition 2.9, an LWE instance is provided as a sampling oracle $\mathcal{O}$ that can be either uniformly random $\mathcal{O}_\$$ or a pseudorandom $\mathcal{O}_s$ for some secret random $\boldsymbol{s} \in \mathbb{Z}_q^n$. The reduction B uses adversary $\mathcal{A}$ to distinguish the two oracles as follows:

Invocation. Reduction $\mathcal{B}$ requests $m + z$ instances from oracle $\mathcal{O}$, i.e. pair $(\boldsymbol{a}_i, b_i)$ for $i = 1, \ldots, m + z$.

Setup. Reduction $\mathcal{B}$ constructs master public key mpk as follows:

1. Set matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ to be the first m vectors $\boldsymbol{a}_i$ in pairs $(\boldsymbol{a}_i, b_i)$ for $i = 1, \ldots, m$.
2. Assign the $\{m + i\}_{i \in [m+1, m+z]}$-th LWE instances $(\boldsymbol{a}_{m+1}^t, \ldots, \boldsymbol{a}_{m+z}^t)$ to be matrix $\mathbf{U} \in \mathbb{Z}_q^{n \times z}$.
3. Construct the reminder of master public key, namely matrices $\{\mathbf{A}_i\}_{i \in [\ell_1]}$ and $\{\mathbf{C}_j\}_{j \in [\ell_2]}$ as in hybrid H_3.
4. Send $\mathsf{mpk} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell_1]}, \{\mathbf{C}_j\}_{j \in [\ell_2]}, \mathbf{U})$ to $\mathcal{A}$.

Queries. Reduction $\mathcal{B}$ answers identity queries as in hybrid H_3, including aborting the simulation if needed.

Challenge. When adversary A sends message $(\boldsymbol{\mu}_0, \boldsymbol{\mu}_1)$ and challenge attribute $(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*)$, reduction $\mathcal{B}$ does the following:

1. Set $\boldsymbol{v} \in \mathbb{Z}_q^m$ the first m integers b_i in LWE pairs $(\boldsymbol{a}_i, b_i)$, for $i = 1, \ldots, m$.
2. Set challenge ciphertext $\mathsf{ct} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$ as

$$\boldsymbol{c}_1 = \left(\boldsymbol{v}, (\mathbf{R}_{\boldsymbol{x}_1^*})^\top \cdot \boldsymbol{v}, (\mathbf{R}_1' | \ldots | \mathbf{R}_{\ell_2}')^\top \cdot \boldsymbol{v}\right)$$
$$\text{and } \boldsymbol{c}_2 = (b_{m+1}, \ldots, b_{m+z}) + \lfloor q/2 \rceil \boldsymbol{\mu}_b.$$

3. Send challenge ciphertext $\mathsf{ct} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$ to adversary $\mathcal{A}$.

Guess. After being allowed to make additional queries, $\mathcal{A}$ guesses if it is interacting with a hybrid H_3 or H_4 challenger. Our simulator outputs the final guess as the answer to the LWE challenge it is trying to solve.

We can see that when $\mathcal{O} = \mathcal{O}_s$, the adversary's view is as in hybrid H_3; when $\mathcal{O} = \mathcal{O}_\$$, the adversary's view is as in hybrid H_4. Hence, $\mathcal{B}$'s advantage in solving LWE is the same as $\mathcal{A}$'s advantage in distinguishing hybrids H_3 and H_4. □

Completing the Proof Recall that $|Q|$ is the upper bound of the number of the adversary's key queries, and ε is the advantage of the adversary in H_0. By Lemmas 2.22 and 2.23, we can know that

$$\Pr_H\left[H(\boldsymbol{x}_1^*) = 0 \bigwedge H(\mathsf{id}_1) \neq 0 \bigwedge \ldots \bigwedge H(\mathsf{id}_{|Q|}) \neq 0\right] \in \left[\frac{1}{q^t}(1 - \frac{Q}{q^t}), \frac{1}{q^t}\right].$$

Thus, we know that for any $(Q+1)$-tuple I denoting a challenge id^* along with ID queries, we have $\varepsilon(I) \in \left(\frac{1}{q^t}(1 - \frac{Q}{q^t}), \frac{1}{q^t}\right)$. Then by setting $[\varepsilon_{min}, \varepsilon_{max}] = \left[\frac{1}{q^t}\left(1 - \frac{Q}{q^t}\right), \frac{1}{q^t}\right]$ in Lemma 4.2, we have

$$\left|\Pr[W_2] - \frac{1}{2}\right| \geq \frac{1}{q^t}\left(1 - \frac{Q}{q^t}\right)\left|\Pr[W_1] - \frac{1}{2}\right| - \frac{Q}{2q^{2t}}.$$

By our parameter setting, $|Q| \leq \frac{1}{2}\varepsilon q^t$, where $\varepsilon = \left|\Pr[W_0] - \frac{1}{2}\right|$, we have that

$$\left|\Pr[W_2] - \frac{1}{2}\right| \geq \frac{1}{q^t}(1 - \frac{Q}{q^t})\left|\Pr[W_0] - \frac{1}{2} - \mathsf{negl}(\kappa)\right| - \frac{Q}{2q^{2t}} \geq \frac{\varepsilon}{4q^t} - \mathsf{negl}(\kappa).$$

We set $t = \lceil \log_q(2|Q|/\varepsilon) \rceil$, then we have $q^t \geq 2|Q|/\varepsilon \geq q^{t-1}$. This implies $\frac{1}{q^t} \geq \frac{\varepsilon}{2q|Q|}$. We can further derive: $\frac{\varepsilon}{4q^t} \geq \frac{\varepsilon^2}{4|Q|q}$. This quantity is non-negligible as long as ε is non-negligible, as q is polynomial for our setting of parameters and $|Q|$ is polynomially bounded.

In summary, as $\Pr[W_4] = \frac{1}{2}$, we have that

$$\begin{aligned}\frac{\varepsilon^2}{4|Q|q} - \mathsf{negl}(\kappa) &\leq \left|\Pr[W_2] - \frac{1}{2}\right| + \mathsf{negl}(\kappa) \\ &\leq \left|\Pr[W_3] - \frac{1}{2}\right| - \mathbf{Adv}_{\mathcal{B}}^{\mathsf{LWE}}(1^\kappa) \\ &\leq \left|\Pr[W_4] - \frac{1}{2}\right| = 0,\end{aligned}$$

which implies $\mathbf{Adv}^{\mathsf{LWE}}_{\mathcal{B}}(1^\kappa) \geq \frac{\varepsilon^2}{4|Q|q} - \mathsf{negl}(\kappa)$. This means the reduction $\mathcal{B}$ defined in Lemma 4.3 breaks the LWE assumption with non-negligible probability. This reaches a contradiction, which completes the proof. □

4.2 ada-sel secure ABE for (t-CNF*) $\wedge_\parallel$ $\mathcal{G}$ from LWE

Before presenting the ABE scheme, let us first recall the building block—conforming cPRF of the ABE construction by Tsabary [40].

Definition 4.4 (*Conforming constrained* PRF [40]) Let $\mathcal{F}$ be a function class such that $\mathcal{F} \subseteq \{0,1\}^\ell \to \{0,1\}$. A conforming constrained PRF for policies in $\mathcal{F}$ is a tuple of PPT algorithms with the following syntax and properties.

- $\mathsf{cPRF.Setup}(1^\kappa) \to (\mathsf{pp}, \mathsf{msk})$ takes as input a security parameter κ and outputs public parameters pp along with a master secret key msk.
- $\mathsf{cPRF.Eval}_{\mathsf{msk}}(x) \to r_x$ is a deterministic algorithm that takes as input a master secret key msk and a bit-string $x \in \{0,1\}^\ell$, and outputs a bit-string $r_x \in \{0,1\}^k$.
- $\mathsf{cPRF.Constrain}_{\mathsf{msk}}(f) \to \mathsf{sk}_f$ takes as input a master secret key msk and a function $f \in \mathcal{F}$, and outputs a constrained key sk_f.
- $\mathsf{cPRF.ConstrainEval}_{\mathsf{sk}_f}(x)$ is a deterministic algorithm that takes as input a constrained key sk_f and a bit-string $x \in \{0,1\}^\ell$, and outputs a bit-string $r'_x \in \{0,1\}^k$.

Correctness A cPRF scheme is correct if for all $x \in \{0,1\}^\ell$ and $f \in \mathcal{F}$ for which $f(x) = 1$, it holds that $\mathsf{cPRF.Eval}_{\mathsf{msk}}(x) = \mathsf{cPRF.ConstrainEval}_{\mathsf{sk}_f}(x)$ where $(\mathsf{pp}, \mathsf{msk}) \leftarrow \mathsf{cPRF.Setup}(1^\kappa)$ and $\mathsf{sk}_f \leftarrow \mathsf{cPRF.Constrain}_{\mathsf{msk}}(f)$.

Gradual evaluation The algorithm cPRF.Constrain (in addition to cPRF.Eval, cPRF.ConstrainEval) is deterministic and the following holds. For any fixing of $\mathsf{pp} \leftarrow \mathsf{cPRF.Setup}(1^\kappa)$, $f \in \mathcal{F}$ and $x \in \{0,1\}^\ell$ for which $f(x) = 1$, define the following circuits:

- $U_{\sigma \to x} : \{0,1\}^\kappa \to \{0,1\}^k$ takes as input msk and ℓ-bits input x, computes
$$r_x = \mathsf{cPRF.Eval}_{\mathsf{msk}}(x).$$
- $U_{\sigma \to f} : \{0,1\}^\kappa \to \{0,1\}^{\ell_f}$ takes as input msk and $f \in \mathcal{F}$, computes
$$\mathsf{sk}_f = \mathsf{cPRF.Constrain}_{\mathsf{msk}}(f).$$
- $U_{f \to x} : \{0,1\}^{\ell_f} \to \{0,1\}^k$ takes as input sk_f and ℓ-bits input x, computes
$$r_x = \mathsf{cPRF.ConstrainEval}_{\mathsf{sk}_f}(x).$$

We require that for all pp, f, x as define above, the circuit $U_{\sigma \to x}$ and the effective sub-circuit of $U_{f \to x} \circ U_{\sigma \to f}$ are the same. That is, the description of $U_{\sigma \to x}$ as a sequence of gates is identical to the sequence of gates that go from the input wires to output wires of circuit $U_{f \to x} \circ U_{\sigma \to f}$.

Pseudorandomness The adaptive security game of a cPRF scheme between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ is as follows.

1. Initialization: $\mathcal{C}$ generates $(\mathsf{pp}, \mathsf{msk}) \to \mathsf{cPRF.Setup}(1^\kappa)$ and sends pp to $\mathcal{A}$.
2. Queries Phase I: $\mathcal{A}$ makes (possibly many) queries in an arbitrary order:

- Evaluation Queries: $\mathcal{A}$ sends a bit-string $x \in \{0,1\}^{\ell}$, $\mathcal{C}$ returns $r_x \leftarrow \mathsf{cPRF.Eval}_{\mathsf{msk}}(x)$.
- Key Queries: $\mathcal{A}$ sends a function $f \in \mathcal{F}$, $\mathcal{C}$ returns

$$\mathsf{sk}_f \leftarrow \mathsf{cPRF.Constrain}_{\mathsf{msk}}(f).$$

3. Challenge Phase: $\mathcal{A}$ sends the challenge bit-string $x^* \in \{0,1\}^{\ell}$. $\mathcal{C}$ uniformly samples $b \xleftarrow{\$} \{0,1\}$. If $b = 0$ then returns $r^* \xleftarrow{\$} \{0,1\}^k$. Otherwise it returns $r^* \leftarrow \mathsf{cPRF.Eval}_{\mathsf{msk}}(x^*)$.
4. Queries Phase II: same as the first queries phase.
5. End of Game: $\mathcal{A}$ outputs a bit b'.

$\mathcal{A}$ wins the game if (1) $b' = b$; (2) all the evaluation queries are not for x^*; and (3) all of the key queries f are such that $f(x^*) = 0$. Moreover, we call it to be single-key adaptive security if in the above described game, $\mathcal{A}$ can only make a single key query throughout the entire game. A cPRF scheme is secure (resp. single-key secure) if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the adaptive (resp. single-key adaptive) security game is at most $1/2 + \mathsf{negl}(\kappa)$.

Key simulation We require a PPT algorithm $\mathsf{KeySim}_{\mathsf{pp}}(f) \to \mathsf{sk}_f$ such that any PPT adversary $\mathcal{A}$ has at most $1/2 + \mathsf{negl}(\kappa)$ probability to win the following game against a challenger $\mathcal{C}$.

- Initialization: $\mathcal{C}$ generates $(\mathsf{pp}, \mathsf{msk}) \leftarrow \mathsf{cPRF.Setup}(1^{\kappa})$ and sends pp to $\mathcal{A}$.
- Evaluation Queries I: $\mathcal{A}$ makes (possible multiple) queries. In each query it sends a bit-string $x \in \{0,1\}^{\ell}$ and $\mathcal{C}$ returns $r_x \leftarrow \mathsf{cPRF.Eval}_{\mathsf{msk}}(x)$.
- Challenge Phase: $\mathcal{A}$ sends the challenge constrain $f^* \in \mathcal{F}$. $\mathcal{C}$ uniformly samples $b \leftarrow \{0,1\}$. If $b = 0$ then $\mathcal{C}$ returns $\mathsf{sk}_{f^*} \leftarrow \mathsf{cPRF.Constrain}_{\mathsf{msk}}(f)$, otherwise, it returns $\mathsf{sk}_{f^*} \leftarrow \mathsf{KeySim}_{\mathsf{pp}}(f)$.
- Evaluation Queries II: same as the first queries phase.
- End of Game: $\mathcal{A}$ outputs a bit b'.

$\mathcal{A}$ wins the game if (1) $b' = b$ and (2) all the evaluation queries x are such that $f^*(x) = 0$.

We first recall a lemma from a prior work, and the present our construction.

Lemma 4.5 [40] *Assuming the hardness of* LWE *with super-polynomial modulo-to-noise ratio, there exists a conforming* cPRF *scheme for* t-CNF *function family such that all the required properties above are satisfied.*

4.2.1 Construction of ABE for (t-CNF*) $\wedge_{\|}$ $\mathcal{G}$

Let $\Pi = (\mathsf{cPRF.Setup}, \mathsf{cPRF.Eval}, \mathsf{cPRF.Constrain}, \mathsf{cPRF.ConstrainEval})$ be a conforming cPRF for t-CNF function family with input length ℓ_1 and output length k, and assume that the length of msk_{Π} is κ. For all $f \in t$-CNF let ℓ_f denote the size of sk_f for the function f. Let $U_{\sigma \to x}$, $U_{\sigma \to f}$ and $U_{f \to x}$ be the circuit as defined in the part of **Gradual Evaluation**, and denote the depth of $U_{f \to x}$ as d_{ce}. Let $\mathcal{G}$ be the function family with input length ℓ_2 and output length 1. For convenience, we denote $\mathcal{F}_2$ as t-CNF* $\wedge_{\|}$ $\mathcal{G}$ for short. $\mathsf{ABE} = (\mathsf{ABE.Setup}_{\mathcal{F}_2}, \mathsf{ABE.Enc}_{\mathcal{F}_2}, \mathsf{ABE.KeyGen}_{\mathcal{F}_2}, \mathsf{ABE.Dec}_{\mathcal{F}_2})$ is defined as follows.

ABE.Setup$_{\mathcal{F}_2}(\mathbf{1}^{\kappa})$: The setup algorithm takes as input a security parameter κ, and then does the following:

1. Sample a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ along with a trapdoor basis $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$ of lattice $\Lambda_q^{\perp}(\mathbf{A})$ by running $\mathsf{TrapGen}$.
2. Sample $(\mathsf{pp}_{\Pi}, \mathsf{msk}_{\Pi}) \leftarrow \mathsf{cPRF.Setup}(1^{\kappa})$, denote $\sigma := \mathsf{msk}_{\Pi}$.

3. Select matrices $\mathbf{B}_1, \ldots, \mathbf{B}_{\ell_1} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
4. Select matrices $\mathbf{C}_1, \ldots, \mathbf{C}_{\ell_2} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
5. Select a random matrix $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n\times z}$.
6. Output the public parameters

$$\mathsf{mpk} = (\mathbf{A}, \{\mathbf{B}_i\}_{i\in[\ell_1]}, \{\mathbf{C}_i\}_{i\in[\ell_2]}, \mathbf{U}, \mathsf{pp}_{\Pi})$$

and the master secret key $\mathsf{msk} = (\mathbf{T}_{\mathbf{A}}, \sigma)$.

ABE.Setup$_{\mathcal{F}_2}$**(mpk, msk,** $U_{\boldsymbol{x}} \wedge_{\|} g$**)**: The key generation algorithm takes as input mpk, msk, a policy function $U_x \wedge_{\|} g \in \mathcal{F}_2$ where the depth of g is d, and then does the following:

1. Compute the matrix $\mathbf{B}_{\sigma\to x} \leftarrow \mathsf{Eval}_{\mathsf{pk}}(U_{\sigma\to x}, \{\mathbf{B}_i\}_{i\in[\kappa]})$, and denote $\mathbf{B}_x = [\mathbf{B}, \ldots, \mathbf{B}_{\ell_1}] \cdot \mathbf{H}_{\sigma\to x}$.
2. Compute $r \leftarrow \Pi.\mathsf{Eval}_{\sigma}(x)$ and let $I_r : \{0,1\}^k \to \{0,1\}$ be the function that on input r' returns 1 if and only if $r = r'$. Compute $\mathbf{B}_r \leftarrow \mathsf{Eval}_{\mathsf{pk}}(I_r, \mathbf{B}_x)$, and denote $\mathbf{B}_{x,r} = \mathbf{B}_x\mathbf{B}_r$.
3. Define function $\bar{g}(\cdot) = 1 - g(\cdot)$, and compute

$$\mathbf{H}_g = \mathsf{Eval}_{\mathsf{pk}}(\bar{g}, \mathbf{C}_1, \ldots, \mathbf{C}_{\ell_2}) \in \mathbb{Z}_q^{n\times m}.$$

4. Let $\mathbf{F}_{U_{x,r}\wedge_{\|}g} = (\mathbf{A}|\mathbf{A}'_{U_{x,r}\wedge_{\|}g}) = (\mathbf{A}|\mathbf{B}_{x,r}|\mathbf{H}_g) \in \mathbb{Z}_q^{n\times 3m}$.
5. Sample $\mathbf{D} \in \mathbb{Z}^{3m\times z}$ as $\mathbf{D} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{A}'_{U_{x,r}\wedge_{\|}g}, \mathbf{U}, \tau)$.
6. Output $\mathsf{sk}_{U_{x,r}\wedge_{\|}g} := (r, \mathbf{D})$, where $\mathbf{F}_{U_{x,r}\wedge_{\|}g} \cdot \mathbf{D} = \mathbf{U} \bmod q$.

ABE.Enc$_{\mathcal{F}_2}$**(mpk,** $(\boldsymbol{f}, \boldsymbol{x})$**,** $\boldsymbol{\mu}$**)**: In order to encrypt a message $\boldsymbol{\mu} \in \{0,1\}^z$ with respect to attribute $(f, \boldsymbol{x})$ where $f \in t$-CNF and $\boldsymbol{x} = (x_1, \ldots, x_{\ell_2}) \in \mathbb{Z}_q^{\ell_2}$, the encryption algorithm first chooses a random vector $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$ and three error vectors $\boldsymbol{e}_0 \leftarrow \chi^m$, $\boldsymbol{e}_1 \leftarrow \tilde{\chi}^{m\cdot\ell_f}$, $\boldsymbol{e}_2 \leftarrow \chi^z$ where χ and $\tilde{\chi}$ are B and $\tilde{B}$ bounded discrete Gaussian distribution, respectively, and then does the following:

1. Sample $\mathsf{sk}_f \leftarrow \mathsf{KeySim}_{\mathsf{pp}}(f)$, and denote $s_f = \mathsf{sk}_f$.
2. Compute $\mathbf{B}_{\sigma\to f} \leftarrow \mathsf{Eval}_{\mathsf{pk}}(U_{\sigma\to f}, \{\mathbf{B}_i\}_{i\in[\kappa]})$, and denote $\mathbf{B}_f = [\mathbf{B}_1, \ldots, \mathbf{B}_{\ell_1}] \cdot \mathbf{B}_{\sigma\to f}$.
3. Set $\mathbf{H}_{\boldsymbol{x}} = (x_1\mathbf{G} + \mathbf{C}_1| \cdots |x_{\ell_2}\mathbf{G} + \mathbf{C}_{\ell_2}) \in \mathbb{Z}_q^{n\times m\ell_2}$.
4. Choose ℓ_2 uniformly random matrices $\mathbf{R}'_j \leftarrow \{-1,1\}^{m\times m}$ for $j \in [\ell_2]$, and set $\boldsymbol{e}_3 = (\mathbf{R}'_1| \cdots |\mathbf{R}'_{\ell_2})^{\top} \cdot \boldsymbol{e}_0 \in \mathbb{Z}_q^{m\ell_2}$.
5. Set $\mathbf{F}_{f,\boldsymbol{x}} = (\mathbf{A}|\mathbf{A}'_{f,\boldsymbol{x}}) = (\mathbf{A}|\mathbf{B}_f - s_f \otimes \mathbf{G}|\mathbf{H}_{\boldsymbol{x}}) \in \mathbb{Z}_q^{n\times(2+\ell_2)m}$.
6. Output $\boldsymbol{c} = (s_f, \mathbf{F}_{f,\boldsymbol{x}}^{\top} \cdot \boldsymbol{s} + (\boldsymbol{e}_0^{\top}, \boldsymbol{e}_1^{\top}, \boldsymbol{e}_3^{\top})^{\top}, \mathbf{U}^{\top} \cdot \boldsymbol{s} + \boldsymbol{e}_2 + \lfloor q/2 \rceil \boldsymbol{\mu})$.

ABE.Dec$_{\mathcal{F}_2}$**(mpk,** $\mathbf{sk}_{U_{x,r}\wedge_{\|}\boldsymbol{g}}$**,** $((\boldsymbol{f}, \boldsymbol{x}), \boldsymbol{c})$**)**: The decryption algorithm uses the key $\mathsf{sk}_{U_{x,r}\wedge_{\|}g} := \mathbf{D}$ to decrypt $\boldsymbol{c}$ with attribute $(f, \boldsymbol{x})$. If $U_{x,r}(f) \neq 1$, output $\perp$. Particularly, let the ciphertext $\boldsymbol{c} = (s_f, \boldsymbol{c}_{in,1}, \boldsymbol{c}_{in,2}, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{\ell_2}, \boldsymbol{c}_{out})$, compute $r' \leftarrow U_{f\to x}(s_f)$. If $r = r'$ then set $U_{x,r}(f) = 0$.

Otherwise, compute $\mathbf{B}_f, \mathbf{B}_x$ as in Enc and KeyGen respectively. Then compute $\mathsf{ct}_{s_f\to r'} \leftarrow \mathsf{Eval}_{\mathsf{ct}}\big(U_{f\to x}, s_f, \mathbf{B}_f\big)$ and $\mathsf{ct}_{r,r'} \leftarrow \mathsf{Eval}_{\mathsf{ct}}\big(I_r, r', \mathbf{B}_x\big)$, and also compute

$$\boldsymbol{c}_g = \mathsf{Eval}_{ct}(\bar{g}, \{(x_i, \mathbf{C}_i, \boldsymbol{c}_i)\}_{i=1}^{\ell_2}).$$

Lastly, output $\boldsymbol{\mu}' = \mathsf{Round}(\mathsf{ct}_{out} - \mathbf{D}^{\top} \cdot (\mathsf{ct}_{in,1}, \mathsf{ct}_{in,2}, \mathsf{ct}_{s_f\to r'}, \mathsf{ct}_{r,r'}, \mathsf{ct}_g))$.

Table 3 Parameter setting of ada-sel secure ABE for $(t\text{-CNF}^*) \wedge_{\|} \mathcal{G}$

Parameters	Description	Setting
κ	Security parameter	
n	PK-lattice row dimension	κ^{ϵ_3}
m	PK-lattice column dimension	$O(n \log q)$
q	Modulus	$B(2n^2)^{3d_{ce}+5}$
d	Depth of g	$O(\log \kappa)$
d_{ce}	Depth of $U_{f\to x}$	κ^{ϵ_2}
τ	SampleLeft and SampleRight parameter	$\kappa(2m)^{d_{ce}+3}$
B	Bound of error distribution χ	$O(\kappa)$
$\tilde{B}$	Bound of error distribution $\tilde{\chi}$	$B\kappa^2(2m)^{d_{ce}+1}$
k	Output length of conforming cPRF	κ
ℓ_2	Input length of g	κ
ℓ_f	The size of sk_f	$O(1)$

Correctness

Lemma 4.6 *If Π is a conforming* cPRF *for function class t-*CNF*, then* $\mathsf{ABE}_{\mathcal{F}_2}$ *is a correct* ABE *scheme for the function class $\mathcal{F}_2$.*

Proof Fix $\boldsymbol{\mu} \in \{0,1\}^z$, $(\mathsf{pp}, \mathsf{msk}) \leftarrow \mathsf{ABE}_{\mathcal{F}_2}.\mathsf{Setup}(1^\kappa)$, $U_{x,r} \wedge_{\|} g \in \mathcal{F}_2$ and attribute $(f, \boldsymbol{x})$ such that $U_{x,r}(f) \wedge g(\boldsymbol{x}) = 1$. Consider the execution of $\mathsf{ABE.Dec}_{\mathcal{F}_2}$.

We can show that $\mathsf{ct}_{r,r'} = \mathbf{B}_r^\top \cdot \boldsymbol{s} + \boldsymbol{e}_1'$ by similar computation as [40], where $\|\boldsymbol{e}_1'\| \le m^2 \ell_f k \tilde{B}(2m)^{d_{ce}+1}$ and $\tilde{B}$ is the bound of distribution $\tilde{\chi}$. On the other hand, $\mathsf{ct}_g = \mathbf{H}_g^\top \cdot \boldsymbol{s} + \boldsymbol{e}_g$, where $\|\boldsymbol{e}_g\| \le 4^d m^{3/2} B$. Therefore,

$$\begin{aligned}
&\mathsf{ct}_{out} - \mathbf{D}^\top \cdot ((\mathsf{ct}_{in,1}, \mathsf{ct}_{in,2}, \mathsf{ct}_{s_f \to r'}, \mathsf{ct}_{r,r'}, \mathsf{ct}_g)) \\
&= \mathbf{U}^\top \cdot \boldsymbol{s} + \boldsymbol{e}_2 + \lfloor q/2 \rceil \boldsymbol{\mu} - \mathbf{D}^\top \cdot (\mathbf{A} | \mathbf{B}_{x,r} | \mathbf{H}_g) \cdot \boldsymbol{s} - \mathbf{D}^\top \cdot (\boldsymbol{e}_0, \boldsymbol{e}_1', \boldsymbol{e}_g) \\
&= \boldsymbol{e}_2 + \lfloor q/2 \rceil \boldsymbol{\mu} - \mathbf{D}^\top \cdot (\boldsymbol{e}_0, \boldsymbol{e}_1', \boldsymbol{e}_g),
\end{aligned}$$

by our choice of parameters, the error term $\tilde{\boldsymbol{e}} = \boldsymbol{e}_2 - \mathbf{D}^\top \cdot (\boldsymbol{e}_0, \boldsymbol{e}_1', \boldsymbol{e}_g)$ satisfies that $\|\tilde{\boldsymbol{e}}\| \le q/4$. This completes the proof of correctness. □

Parameter setting for this construction For arbitrarily small constant $\epsilon_1 \in (0,1)$ and constant ϵ_2, we denote $\epsilon_3 = \frac{2\epsilon_2}{\epsilon_1}$, and set the system parameters according to the Table 3 below.

These values are chosen in order to satisfy the following constraints:

- To ensure correctness, we require $\|\boldsymbol{e}_2 - \mathbf{D}^\top \cdot (\boldsymbol{e}_0, \boldsymbol{e}_1', \boldsymbol{e}_g)\|_\infty \le q/4$; here we bound the dominating term:

$$\|\mathbf{D}^\top \cdot \boldsymbol{e}_1'\|_\infty \le \tau \sqrt{3m} \cdot m^2 \ell_f k \tilde{B}(2m)^{d_{ce}+1} \le q/4.$$

- For SampleLeft, we know $\|\widetilde{\mathbf{T}_\mathbf{A}}\| = O(\sqrt{n \log q})$, so require that the sampling width τ satisfies

$$\tau \ge O(\sqrt{n \log q}) \cdot \omega(\sqrt{\log 3m}).$$

– For SampleRight. we know $\|\widetilde{\mathbf{T}_\mathbf{G}}\| \leq \sqrt{5}$ and that

$$\tau \geq \sqrt{5} \cdot m^2 \kappa (2m)^{d_{ce}+1} \cdot \omega(\sqrt{\log m}) \geq \|\widetilde{\mathbf{T}_\mathbf{G}}\| \cdot s_{\mathbf{R}_{\sigma \to r}} \cdot \omega(\sqrt{\log m}).$$

– To apply Regev's reduction, we need $B \geq \sqrt{n}\omega(\log n)$.
– To apply the Leftover Hash Lemma, we need $m \geq (n+1)\log q + \omega(\log n)$.

Secret key size We give a simple analysis of the secret key size of our $\mathsf{ABE}_{\mathcal{F}_2}$ construction. By Lemma 2.2, we know that

$$\Pr[\mathbf{D} \leftarrow D_{\Lambda_q^{\mathbf{U}}(\mathbf{F}_{U_{x^\wedge \|g}}),\tau} : \|\boldsymbol{D}\| > \tau\sqrt{3m}] \leq \mathsf{negl}(n).$$

By our setting of parameters above, the size of the secret key of our ABE scheme for $\mathcal{F}_2$ is bounded by $O(\kappa^{2\epsilon_3+\epsilon_2}\log^2\kappa)$.

4.2.2 Security proof of $\mathsf{ABE}_{\mathcal{F}_2}$

Theorem 4.7 *For parameter setting in Table 3,* $\mathsf{ABE}_{\mathcal{F}_2}$ *scheme above is ada-sel secure as defined in Definition 3.5 and Remark 3.6, assuming the* $\mathsf{LWE}_{n,q,\chi}$ *assumption holds.*

Proof The proof proceeds in a sequence of games where the first game is identical to the security experiment as in Definition 3.5, while in the last game in the sequence the adversary has advantage zero. Our goal is to prove indistinguishability among the adjacent games. We let W_i denote the event that adversary wins the $\mathsf{ABE}_{\mathcal{F}_2}$ security experiment in game i, thus adversary's advantage in game i is $|\Pr[W_i] - 1/2|$. The sequence of games can be described as follows:

The sequence of hybrids ($\mathsf{H_0}$, $\mathsf{H_1}$, $\mathsf{H_2}$, $\mathsf{H_3}$, $\mathsf{H_4}$)

Hybrid H_0: This is the original security experiment $\mathbf{Exp}_{\mathcal{A}}^{\text{ada-sel}}(1^\kappa)$ from Definition 3.5 between the adversary $\mathcal{A}$ and the challenger.
Hybrid H_1: Hybrid H_1 is identical to Hybrid H_0 except that we add an abort event that is independent of the adversary's view. Suppose the number of queries made by adversary is Q which is a polynomial in κ. And let $(x_1, g_1), \cdots, (x_Q, g_Q)$ denote the key queries. W.l.o.g., assume that there does not exist one query (x_i, g_i) such that $f^*(x_i) = g_i(x^*) = 1$, where f^* and x^* are the first and the second part of the challenge attribute.
In final guess phase, upon receiving the adversary's guess $b' \in \{0,1\}$ for b, the challenger does the abort check: $f^*(x_i) \neq 1$ and $g_i(x^*) \neq 1$. If the condition does not hold, the challenger overwrites b' with a freshly random bit in $\{0,1\}$, and we say the challenger aborts the game.
Hybrid H_2: We change the way challenger generates the challenge ciphertext. Instead of computing $\mathsf{sk}_{f^*} \leftarrow \mathsf{KeySim}_{\mathsf{pp}}(f^*)$, it computes

$$\mathsf{sk}_{f^*} \leftarrow \mathsf{cPRF.Constrain}_{\mathsf{msk}}(f^*).$$

Now $\mathsf{sk}_{f^*} = U_{\sigma \to f^*}(\sigma)$.
Hybrid H_3: We change the way challenger generates the matrices $\{\mathbf{B}_i\}, \{\mathbf{C}_j\}$ as follows. It samples uniformly random matrices $\{\mathbf{R}_i\}, \{\mathbf{R}'_j\}$, where $\mathbf{R}_i \xleftarrow{\$} \{0,1\}^{m\times m}$, $\mathbf{R}'_j \xleftarrow{\$} \{0,1\}^{m\times m}$, and set $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i + \sigma_i\mathbf{G}$, $\mathbf{C}_j = \mathbf{A}\mathbf{R}'_j - x_{2j}\mathbf{G}$.
Hybrid H_4: We change the way challenger generates the challenge ciphertext again. Specifically, let the ciphertext $\boldsymbol{c} = (s_f, \boldsymbol{c}_{in,1}, \boldsymbol{c}_{in,2}, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{\ell_2}, \boldsymbol{c}_{out})$. Recall that previously $\boldsymbol{c}_{in,2} = \boldsymbol{s}^\top(\mathbf{B}_f - s_f \otimes \mathbf{G}) + \boldsymbol{e}_1^\top$, $\boldsymbol{c}_j = \boldsymbol{s}^\top(x_j\mathbf{G} + \mathbf{C}_j) + \boldsymbol{e}_{3,j}^\top$, where $j \in [\ell_2]$, $\boldsymbol{e}_1 \xleftarrow{\$}$

$\tilde{\chi}^{m\cdot\ell_f}$. In this hybrid, these vectors will be computed as $\boldsymbol{c}_{in,2} = \boldsymbol{c}_{in,1}\cdot\mathbf{R}_{\sigma\to f} + \boldsymbol{e}_1^\top$, where $\mathbf{R}_{\sigma\to f} = \mathsf{Eval}_{\mathsf{Sim}}(U_{\sigma\to f}, \{(\sigma_i, \mathbf{R}_i)\}_{i=1}^{\kappa}, \mathbf{A})$, and $\boldsymbol{c}_j = \boldsymbol{c}_{in,1}\cdot\mathbf{R}'_j$.
Hybrid H_5: We change the way challenger answers key queries. Let x be a query and fix $r' \leftarrow \mathsf{Eval}_\sigma(x)$. Note that $\mathbf{B}_{\sigma\to x} = \mathbf{A}\mathbf{R}_{\sigma\to x} + r\otimes\mathbf{G}$, where $\mathbf{R}_{\sigma\to x} = \mathsf{Eval}_{\mathsf{Sim}}(U_{\sigma\to x}, \{(\sigma_i, \mathbf{R}_i)\}_{i=1}^{\kappa}, \mathbf{A})$, and $\mathbf{B}_r = \mathbf{A}\mathbf{R}_r + I_r(r')\otimes\mathbf{G}$, where $\mathbf{R}_r = \mathsf{Eval}_{\mathsf{Sim}}(I_r, (r, \mathbf{R}_{\sigma\to x}), \mathbf{A})$.
Since $U_x(f^*)\wedge g(\boldsymbol{x}^*) = 0$, then $\Pr[\neg(f^*(x) = 1\wedge g(\boldsymbol{x}^*) = 1)] = 1$. If $f^*(x) = 1$, then $I_r(r') = 0$ with overwhelming probability. On the other hand, when $f^*(x) = 1$, $g(\boldsymbol{x}^*)$ must be 0, then $\mathbf{H}_g = \mathbf{A}\mathbf{R}'_g + (1 - g(\boldsymbol{x}^*))\mathbf{G} = \mathbf{A}\mathbf{R}'_g + \mathbf{G}$, where $\mathbf{R}'_g = \mathsf{Eval}_{\mathsf{Sim}}(\bar{g}, (\boldsymbol{x}^*, \{\mathbf{R}'_j\}), \mathbf{A})$. Now challenger can use algorithm $\mathsf{SampleRight}$ to make the following equation hold

$$[\mathbf{A}|\mathbf{A}\mathbf{R}_r|\mathbf{A}\mathbf{R}'_g + \mathbf{G}]\cdot\mathbf{D} = \mathbf{U} \bmod q.$$

Similarly, If $f^*(x) = 0$, $I_r(r') = 1$. Then challenger can also use algorithm $\mathsf{SampleRight}$ to make the following equation hold

$$[\mathbf{A}|\mathbf{A}\mathbf{R}_r + \mathbf{G}|\mathbf{A}\mathbf{R}'_g + (1 - g(\boldsymbol{x}^*))\mathbf{G}]\cdot\mathbf{D} = \mathbf{U} \bmod q,$$

no matter $g(\boldsymbol{x}^*) = 0$ or 1.
Hybrid H_6: We change the way $\mathbf{A}$ is generated. Instead of sampling it via $\mathsf{TrapGen}$, we sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ uniformly at random.
Hybrid H_7: We change again the way challenger generates the challenge ciphertext. It now samples $\boldsymbol{c}_{in,1}$ and $\boldsymbol{c}_{out}$ uniformly at random. Now the challenge completely hides b and so adversary has no advantage.

Now we explain the indistinguishability between the adjacent hybrids briefly. For H_0 and H_1, the challenger in H_1 has probability $\frac{1}{Q}$ that doesn't abort the game, so $|\Pr[W_1] - 1/2| = \frac{1}{Q}|\Pr[W_0] - 1/2|$. The indistinguishability between H_1 and H_2 comes from the pseudorandomness of the underling PRF of the cPRF. H_2 is indistinguishable from H_3 because of the **Key Simulation** security and the fact that random r_i doesn't leak any information of msk. We can apply the leftover hash Lemma 2.6 to show the indistinguishability between H_3 and H_4. H_4 is indistinguishable from H_5 due to the smudging Lemma 2.8. The indistinguishability between H_5 and H_6 comes from Lemma 2.3. H_6 is indistinguishable from H_7 because of Lemma 2.1. Finally, H_7 is indistinguishable from H_8 due to the hardness of LWE.

In conclusion, $|\Pr[W_0] - 1/2| = Q|\Pr[W_1] - 1/2| \le Q(|\Pr[W_2] - 1/2| + \varepsilon_{\mathsf{PRF}}) \le Q(|\Pr[W_3] - 1/2| + \varepsilon_{\mathsf{PRF}} + \varepsilon_{\mathsf{keysim}}) \le \cdots \le Q(|\Pr[W_8] - 1/2| + \varepsilon_{\mathsf{PRF}} + \varepsilon_{\mathsf{keysim}} + \varepsilon_{\mathsf{LWE}} + \mathsf{negl}(\kappa)) = Q(\varepsilon_{\mathsf{PRF}} + \varepsilon_{\mathsf{keysim}} + \varepsilon_{\mathsf{LWE}} + \mathsf{negl}(\kappa))$. Therefore, the advantage of adversary in ABE security game is negligible assuming the security of cPRF and the hardness of LWE. □

As a direct corollary of this section, we obtain the following $\mathsf{AB\text{-}wHPS}$ from lattices.

Corollary 4.8 *Assuming* LWE*, there exists* $\mathsf{AB\text{-}wHPS}$ *that is*

1. *adaptively secure for the comparison functions;*
2. *adaptively secure for* t*-*CNF^* *functions.*
3. *selectively secure for general circuits.*

Moreover, the secret key size (excluding the function description) of the $\mathsf{AB\text{-}wHPS}$ *only depends on the depth of the function, but not the size.*

5 Optimal-rate leakage-resilient encryption schemes in the relative leakage model

Prior work (e.g., Naor and Segev [34], Alwen et al. [6], and Hazay et al. [26]) showed how to construct leakage resilient PKE/IBE from wHPS/IB-wHPS in the relative model. The construction can be generalized to construct leakage resilient ABE from AB-wHPS in the same spirit. To further achieve the optimal leakage rate, we observe that all we need is an AB-wHPS with succinct keys (which do not depend on the function size). This is what we construct in Sect. 3.2, i.e., Construction 3.11, Theorem 3.12, AB-wHPS and the underlying ABE instantiations in Corollary 4.8.

Construction 5.1 *Let* Π =AB-wHPS.{Setup, KeyGen, Encap, Encap*, Decap} *be a* $(\log|\mathcal{K}|, \log|\mathcal{K}|)$*-universal* AB-wHPS *with the encapsulated-key-space* $\mathcal{K}$ *and attribute space* $\mathcal{X} = \{0,1\}^*$ *for a class of policy functions* $\mathcal{F} = \{f : \{0,1\}^* \to \{0,1\}\}$. *Let* $\mathsf{Ext} : \mathcal{K} \times \mathcal{S} \to \mathcal{M}$ *be a* $(\log|\mathcal{K}| - \ell, \varepsilon)$*-extractor, where three sets* $\mathcal{K}, \mathcal{S}, \mathcal{M}$ *are efficient ensembles,* $\ell = \ell(\kappa)$ *is some parameter and* $\varepsilon = \varepsilon(\kappa) = \mathsf{negl}(\kappa)$ *is negligible. Furthermore, assume that* $\mathcal{M}$ *is an additive group. Then, a leakage-resilient* ABE *scheme* $\Pi_{\mathcal{F}} = \Pi_{\mathcal{F}}$.{Setup, KeyGen, Enc, Dec} *with message space* $\mathcal{M}$ *and policy function class* $\mathcal{F}$ *can be constructed as follows:*

- $\Pi_{\mathcal{F}}.\mathsf{Setup}(1^\kappa)$*: The algorithm runs* $(\mathsf{mpk}^{\Pi}, \mathsf{msk}^{\Pi}) \xleftarrow{\$} \Pi.\mathsf{Setup}(1^\kappa)$*, and outputs* $\mathsf{mpk} := \mathsf{mpk}^{\Pi}$*, and* $\mathsf{msk} := \mathsf{msk}^{\Pi}$.
- $\Pi_{\mathcal{F}}.\mathsf{KeyGen}(\mathsf{msk}, f)$*: Given a master secret-key* msk *and a function* $f \in \mathcal{F}$ *as input, the algorithm runs* AB-wHPS.KeyGen *to generate and output* (f, sk_f^{Π})*, where* $\mathsf{sk}_f := \mathsf{sk}_f^{\Pi} \xleftarrow{\$} \mathsf{AB\text{-}wHPS.KeyGen}(\mathsf{msk}, f)$.
- $\Pi_{\mathcal{F}}.\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x}, \mu)$*: Given a master public-key* mpk*, an attribute* $\boldsymbol{x} \in \mathcal{X} = \{0,1\}^*$*, and a message* $\mu \in \mathcal{M}$ *as input, the algorithm runs* AB-wHPS.Encap *to generate* $(CT', k) \leftarrow \mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x})$*, and then samples* $s \xleftarrow{\$} \mathcal{S}$*. Furthermore, the algorithm computes and outputs*

$$ct = (s, ct_0, ct_1) = (s, CT', \mu + \mathsf{Ext}(k, s)).$$

- $\Pi_{\mathcal{F}}.\mathsf{Dec}(\mathsf{sk}_f, ct)$*: Given a ciphertext* $ct = (s, ct_0, ct_1)$ *and a secret key* sk_f *as input, the algorithm runs* AB-wHPS.Decap *to generate* $k = \mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, ct_0)$*, and then output* $\mu = ct_1 - \mathsf{Ext}(k, s)$.

Our construction achieves a leakage resilient ABE, and can be modified into a leakage resilient PKE/IBE. We summarize the results in the following theorem.

Theorem 5.2 *Assume* Π *is a selectively (or adaptively, resp.) secure* $(\log|\mathcal{K}|, \log|\mathcal{K}|)$*-universal* AB-wHPS *for the policy function class* $\mathcal{F}$*, and* $\mathsf{Ext} : \mathcal{K} \times \mathcal{S} \to \mathcal{M}$ *be a* $(\log|\mathcal{K}| - \ell, \mathsf{negl}(\kappa))$*-extractor. Then the above* ABE *scheme* $\Pi_{\mathcal{F}} = \Pi_{\mathcal{F}}$.{Setup, KeyGen, Enc, Dec} *for* $\mathcal{F}$ *is a selectively (or adaptively, resp.)* $\ell(\kappa)$*-leakage resilient attribute-based encryption scheme for the policy function class* $\mathcal{F}$ *in the relative-leakage model. Particularly,* $\Pi_{\mathcal{F}}$ *is aslo*

- *an* $\ell(\kappa)$*-leakage-resilient* PKE *in the relative-leakage model, if* $\mathcal{F}$ *contains only a single function that always outputs* 1.
- *an* $\ell(\kappa)$*-leakage-resilient* IBE *in the relative-leakage model, if* $\mathcal{F}$ *contains the following comparison functions, i.e., each function* $f_{\boldsymbol{y}} \in \mathcal{F}$ *is indexed by a vector* $\boldsymbol{y}$*, and* $f_{\boldsymbol{y}}(\boldsymbol{x}) = 1$ *if and only if* $\boldsymbol{y} = \boldsymbol{x}$.

Proof Here, we just prove the general case of ABE for a general function class $\mathcal{F}$. Then, the results for IBE and PKE are clearly set up, since IBE and PKE are special cases of ABE for equation-testing functions and constant function, respectively.

First, the correctness of this ABE scheme $\Pi_{\mathcal{F}}$ follows naturally from that of AB-wHPS Π. Furthermore, the security of this ABE scheme can be argued by using a sequence of hybrids as follows.

Hybrid $\mathbf{H_0}$: This hybrid is defined to be the security experiment with ℓ-leakage in Definition 2.11. In this hybrid, the view of $\mathcal{A}$ consists of the master public-key pk, leakage information $h(\mathsf{sk}_f)$, and challenge ciphertext $(s, \mathsf{CT}_0, \mathsf{CT}_1)$, where $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Setup}(1^\kappa)$, $\mathsf{sk}_f \xleftarrow{\$} \mathsf{AB\text{-}wHPS.KeyGen}(\mathsf{msk}, f)$, $s \xleftarrow{\$} \mathcal{S}$,

$$(\mathsf{CT}_0, k) \leftarrow \mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x}), \quad \mathsf{CT}_1 = \mu_b + \mathsf{Ext}(k, s).$$

Notice that the leakage function $h : \{0,1\}^* \to \{0,1\}$ is chosen adaptively by the adversary before the challenge stage. More importantly, in the leakage query stage, $\mathcal{A}$ is allowed to query only one policy function f such that $f(x^*) = 1$ where x^* is the challenge attribute.

Hybrid $\mathbf{H_1}$: This hybrid is almost identical to the Hybrid 0, except the challenge ciphertext is computed in the following way:

$$(\mathsf{CT}_0, k) \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x}), \quad k_1 = \mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, \mathsf{CT}_0),$$
$$\mathsf{CT}_1 = \mu_b + \mathsf{Ext}(k_1, s).$$

The only difference between Hybrid 0 and Hybrid 1 is the usage of k and k_1 in the computation of c_1. In fact, $k = k_1$ according to the correctness of the underlying AB-wHPS. Hence, Hybrid 0 and Hybrid 1 are identical.

Hybrid $\mathbf{H_2}$: This hybrid is almost same to Hybrid 1, except the challenge ciphertext is computed in the following way:

$$\mathsf{CT}'_0 \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x}), \quad k_1 = \mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, \mathsf{CT}'_0),$$
$$\mathsf{CT}_1 = \mu_b + \mathsf{Ext}(k_1, s).$$

The only difference between Hybrid 1 and Hybrid 2 is the computation and usage of CT_0 and CT'_0. In fact, according to the ciphertext indistinguishability of the underlying AB-wHPS, CT_0 and CT'_0 are computationally indistinguishable even for an adversary having secret key sk_f. Hence, Hybrid 0 and Hybrid 1 are indistinguishable for an adversary having the leakage information $h(\mathsf{sk}_f)$. Notice that, in the real scenarios, one party is always issued just one secret key satisfying his attributes, which will be used in the following decryption computation. Therefore, it makes sense for us to limit just one policy function f such that $f(x^*) = 1$ in the leakage query stage.

Hybrid $\mathbf{H_3}$: This hybrid is almost same to Hybrid 2, except that the challenge ciphertext is computed in the following way:

$$\mathsf{CT}'_0 \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}^*(\mathsf{mpk}, \boldsymbol{x}), \quad r \xleftarrow{\$} \mathcal{M}, \quad \mathsf{CT}_1 = \mu_b + r.$$

Essentially, pk, CT'_0, $k_1 = \mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, \mathsf{CT}'_0)$ and $h(\mathsf{sk}_f)$ are correlated variables. According to the universality of underlying AB-wHPS, we know that k_1 is uniform over $\mathcal{K}$ even given pk and CT'_0, i.e.,

$$H_\infty(k_1 | \mathsf{pk}, \mathsf{CT}'_0) = \log(|\mathcal{K}|).$$

Furthermore, since the bit-length of leakage information $h(\mathsf{sk}_f)$ is ℓ, we have

$$H_\infty(k_1|\mathsf{pk}, \mathsf{CT}'_0, h(\mathsf{sk}_f)) \geq \log(|\mathcal{K}|) - \ell.$$

Then, for a random $s \xleftarrow{\$} \mathcal{S}$, $\mathsf{Ext}(k_1, s)$ is ε-close to the uniform distribution over $\mathcal{M}$ even given $\mathsf{pk}, \mathsf{CT}'_0, h(\mathsf{sk}_f)$, since Ext is assumed to be a strong $(\log(|\mathcal{K}|) - \ell, \varepsilon)$-extractor for $\varepsilon = \mathsf{negl}(\kappa)$. As a result, Hybrid 2 and Hybrid 3 are statistically indistinguishable.

Notice that the view of $\mathcal{A}$ in Hybrid 3 is completely independent of μ_b and b. Therefore, the advantage of $\mathcal{A}$ in Hybrid 3 is 0. Finally, combining all above hybrids together, we conclude that the advantage of $\mathcal{A}$ in Hybrid 0 is also negligible in κ. Thus the ABE scheme $\Pi_{\mathcal{F}}$ is ℓ-leakage-resilient for $\mathcal{F}$. □

Combining Theorems 3.12 and 5.2, we obtain the following results. Assume there exists a sel-sel (or ada-sel) secure ABE scheme with the message space $\mathbb{Z}_m$ for the function class $\mathcal{F} \wedge_{\|} \mathcal{G}$, where $\mathcal{G}$ is the class as in Definition 3.9 with parameters m, n, and the key-length (of the extra part, excluding the function description of f) of this underlying ABE scheme for policy function f is $s(f)$. Then the allowed leakage length of the above leakage resilient ABE (or IBE or PKE) scheme $\Pi_{\mathcal{F}}$ for the function class $\mathcal{F}$ is $\ell = (n \log m - 2\kappa)$ and the key-length of $\Pi_{\mathcal{F}}$ for f is $|\mathsf{sk}_f| = n \log m + |f| + s(\hat{f}_{f,g_{\boldsymbol{y}}})$.

Furthermore, if the secret key size $s(\hat{f}_{f,g_{\boldsymbol{y}}})$ is succinct, i.e., $s(\hat{f}_{f,g_{\boldsymbol{y}}}) = o(|\hat{f}_{f,g_{\boldsymbol{y}}}|) = o(n \log m + |f|)$, then we can set sufficiently large n, m such that $n \log m = \omega(|f|)$. Consequently, the leakage rate of this scheme $\Pi_{\mathcal{F}}$ is $\frac{n \log m - 2\kappa}{n \log m + |f| + s(\hat{f}_{f,g_{\boldsymbol{y}}})} = \frac{1 - \frac{2\kappa}{n \log m}}{1 + \frac{s(\hat{f}_{f,g_{\boldsymbol{y}}}) + |f|}{n \log m}} \approx 1 - o(1)$, achieving the desired optimal leakage rate.

Finally, by combining Corollary 4.8 and Theorem 5.2, we obtain the following Corollary.

Corollary 5.3 *Assuming* LWE, *for all polynomial* $S = \mathsf{poly}(\kappa)$, *there exist* $1 - o(1)$ *leakage resilient* ABE *schemes in the relative leakage model, which are*

1. *adaptively secure for the comparison functions;*
2. *adaptively secure for* t-CNF* *functions of size up to* S*;*
3. *selectively secure for general circuits of size up to* S.

Remark 5.4 We note that our ABE schemes are leakage resilient even if the policy function goes beyond the size bound S. The leakage rate would still be $1 - o(1)$ for a slightly restricted class that leaks $n \log m - 2\kappa$ on the part $\boldsymbol{y}$, the whole description of f, and the extra part of sk_f^{Π} (excluding the function description) of the underlying AB-wHPS. This is more restrictive than functions that leak $n \log m - 2\kappa + |f|$ from the whole secret key.

6 Extension I: optimal-rate leakage-resilient encryption schemes in the BRM

In this section, we present how to use AB-wHPS to construct optimal-rate leakage resilient ABE in the BRM. We follow the structure of [6, 26] by first amplifying the hash proof system and then combining it with a locally computable extractor [41]. In particular, we first amplify AB-wHPS through parallel repetition and random sampling in Sect. 6.1. Then, in Sect. 6.2, we generalize the notion of locally computable extractor by Vadhan [41] into one with larger alphabets, and show that a refined analysis of this tool can be used to derive $1 - o(1)$ leakage rate in the BRM, improving the prior analysis [6, 36] that can only achieve a constant leakage

rate. Finally in Sect. 6.3, we present the overall construction of our leakage resilient ABE in the BRM with optimal leakage rate.

6.1 Amplification of AB-wHPS

Definition 6.1 Let n' be a positive integer, and $\mathcal{H} = \{h : [n'] \to \{0,1\}\}$ be a function class where each function $h_y \in \mathcal{H}$ is indexed by a value $y \in [n']$, and $h_y(x) = 1$ if and only if $x = y$.[9]

Construction 6.2 (Construction of amplified AB-wHPS) *Let* $\Pi = \Pi.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Encap}^*, \mathsf{Decap}\}$ *be an* AB-wHPS *with the encapsulated-key-space* $\mathcal{K}$ *and attribute space* $\mathcal{X} = \{0,1\}^* \times [n']$ *for a class of functions* $\mathcal{F} \wedge_{\|} \mathcal{H}$*, and let* $t \leq n'$ *be a positive integer. Then a new* AB-wHPS $\Pi_{\|}^{n',t}$ *with attribute space* $\{0,1\}^*$ *and encapsulated-key-space* $\mathcal{K}^t$ *for the function class* $\mathcal{F}$ *can be constructed as follows.*

- $\Pi_{\|}^{n',t}.\mathsf{Setup}(1^\kappa)$*: The algorithm runs* $(\mathsf{mpk}^\Pi, \mathsf{msk}^\Pi) \xleftarrow{\$} \Pi.\mathsf{Setup}(1^\kappa)$*, and outputs* $\mathsf{mpk} := \mathsf{mpk}^\Pi$*, and* $\mathsf{msk} := \mathsf{msk}^\Pi$.
- $\Pi_{\|}^{n',t}.\mathsf{KeyGen}(\mathsf{msk}, f)$*: Given a function* $f \in \mathcal{F}$*, the algorithm first sets* $\hat{f}^i = \hat{f}^i_{f,h_i} \in \mathcal{F} \wedge_{\|} \mathcal{H}$ *for every* $i \in [n']$*, and runs* Π n' *times to generate* $\mathsf{sk}_{\hat{f}^i} \xleftarrow{\$} \Pi.\mathsf{KeyGen}(\mathsf{msk}^\Pi, \hat{f}^i)$ *for* $i \in [n']$*. The algorithm outputs*

$$\mathsf{sk}_f := \left(\mathsf{sk}_{\hat{f}^1}, \mathsf{sk}_{\hat{f}^2}, \ldots, \mathsf{sk}_{\hat{f}^{n'}}\right).$$

- $\Pi_{\|}^{n',t}.\mathsf{Encap}(\mathsf{mpk}, \boldsymbol{x})$*: Given* mpk *and an attribute* $\boldsymbol{x} \in \{0,1\}^*$ *as input, the algorithm chooses a random subset* $\boldsymbol{r} := \{r_1, \ldots, r_t\} \subseteq [n']$ *and computes*[10]

$$(CT_i, k_i) \xleftarrow{\$} \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_i)) \text{ for all } i \in [t].$$

  The algorithm finally outputs $CT := (\boldsymbol{r}, CT_1, \ldots, CT_t)$ *and* $\boldsymbol{k} = (k_1, \ldots, k_t)^\top$.
- $\Pi_{\|}^{n',t}.\mathsf{Encap}^*(\mathsf{mpk}, \boldsymbol{x})$*: Given* mpk *and an attribute* $\boldsymbol{x} \in \{0,1\}^*$ *as input, the algorithm chooses a random subset* $\boldsymbol{r} := \{r_1, \ldots, r_t\} \subseteq [n']$ *and computes*

$$CT_i \xleftarrow{\$} \Pi.\mathsf{Encap}^*(\mathsf{mpk}, (\boldsymbol{x}, r_i)) \text{ for all } i \in [t].$$

  Finally, the algorithm outputs $CT := (\boldsymbol{r}, CT_1, \ldots, CT_t)$.
- $\Pi_{\|}^{n',t}.\mathsf{Decap}(\mathsf{sk}_f, CT)$*: Given a ciphertext* $CT := (\boldsymbol{r}, CT_1, \ldots, CT_t)$ *and a secret key* $\mathsf{sk}_f := \left(\mathsf{sk}_{\hat{f}^1}, \mathsf{sk}_{\hat{f}^2}, \ldots, \mathsf{sk}_{\hat{f}^{n'}}\right)$*, the algorithm runs* $\Pi.\mathsf{Decap}$ *to generate* $k_i = \Pi.\mathsf{Decap}(\mathsf{sk}_{\hat{f}^{r_i}}, CT_i)$ *for* $i \in [t]$*, and outputs* $\boldsymbol{k} = (k_1, \ldots, k_t)^\top$ *if* $\hat{f}^{r_i}(\boldsymbol{x}, r_i) = 1$ *for all* $i \in [t]$*. Otherwise, the algorithm outputs* $\perp$.

Next, we present the following amplification theorem, which is essential an extension of the work [6].

[9] Clearly, the domain of h_y is $[n']$. And the parameter n', whose concrete setting is described in Sect. 6.3, is set to achieve the optimal leakage rate for the encryption in the bounded-retrieval model.

[10] The subset $\{r_1, \ldots, r_t\}$ must be randomly chosen, as it is an important property for the analysis of locally computable extractor in Sect. 6.2.

Theorem 6.3 *Assume Π is an (l, w)-universal* AB-wHPS *with the encapsulated-key-space $\mathcal{K}$ for $\mathcal{F} \wedge_{\|} \mathcal{H}$. Then the above amplified construction of $\Pi_{\|}^{n',t}$ is an $(t \cdot l, t \cdot w)$-universal* AB-wHPS *with the encapsulated-key-set $\mathcal{K}^t$ for $\mathcal{F}$. Furthermore,*

- *if the underlying Π is selectively (or adaptively) secure, then the $\Pi_{\|}^{n',t}$ is also selectively (or adaptively) secure;*
- *if the secret-key-size of Π scheme for the policy function f is $(|f| + s(f))$,[11] then the secret-key size of the $\Pi_{\|}^{n',t}$ for f is $n' \times (|f| + \log n' + s(\hat{f}_{f,h}))$.*

Proof The second part of the theorem follows directly by our construction from the underlying Π to the amplified $\Pi_{\|}^{n',t}$, especially by the relationship between policy functions of Π and that of $\Pi_{\|}^{n',t}$.

Similar to Theorem 3.12, in order to prove the first part of this theorem, we need to prove the following three properties: correctness, smoothness and ciphertext indistinguishability.

Correctness Correctness of our $\Pi_{\|}^{n',t}$ follows directly from the correctness of the underlying Π.

Universality As $\Pi_{\|}^{n',t}$ is a parallel repetition of the underlying Π, universality of our $\Pi_{\|}^{n',t}$ follows directly from the universality of the underlying Π.

Ciphertext indistinguishability We prove that the ciphertexts output by $\Pi_{\|}^{n',t}.\mathsf{Encap}(\mathsf{mpk}, \boldsymbol{x}^*)$ and $\Pi_{\|}^{n',t}.\mathsf{Encap}^*(\mathsf{mpk}, \boldsymbol{x}^*)$ are indistinguishable, given one secret "1-key" sk_f such that $f(\boldsymbol{x}^*) = 1$ and perhaps many "0-keys" $\mathsf{sk}_{f'}$ such that $f'(\boldsymbol{x}^*) = 0$, where $\boldsymbol{x}^*$ is the challenge attribute. We summarize the result in the lemma below. □

Lemma 6.4 (Ciphertext indistinguishability) *The construction of the amplified* AB-wHPS *satisfies valid/invalid cipheretext indistinguishability as Definition 3.1.*

Proof We prove the valid/invalid ciphertext indistinguishability of AB-wHPS via a hybrid argument. More specifically, we define the following hybrids, where we start from a valid ciphertext, and then switch row-by-row towards an invalid ciphertext. We prove that each two neighboring hybrids are indistinguishable via a reduction from the underlying AB-wHPS. The proof of this lemma follows directly from the indistinguishability of these hybrids.

Hybrid $\mathbf{H}_0$: For a randomly chosen subset $\boldsymbol{r} := \{r_1, \ldots, r_t\} \subseteq [n']$, this hybrid is defined as the ciphertext indistinguishability experiment in Definition 3.1, where $\mathcal{A}$ is given a valid ciphertext

$$\mathsf{CT}_0 := (\boldsymbol{r}, \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_1)), \ldots, \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_t))),$$

In this hybrid, it is clear that the ciphertext CT_0 is generated as $\Pi_{\|}^{n',t}.\mathsf{Encap}$.

Hybrid $\mathbf{H}_z$: For any $1 \le z \le t-1$, H_z is almost same to H_{z-1}, except that $\mathcal{A}$ is given the following ciphertext

$$\begin{aligned}\mathsf{CT}_z :=& (\boldsymbol{r}, \Pi.\mathsf{Encap}^*(\mathsf{mpk}, (\boldsymbol{x}, r_1)), \ldots, \Pi.\mathsf{Encap}^*(\mathsf{mpk}, (\boldsymbol{x}, r_z)),\\ & \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_{z+1})) \ldots, , \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_t))).\end{aligned}$$

In this hybrid, the first z ciphertexts are generated by $\Pi.\mathsf{Encap}^*$ (with z different attributes), and the rest are by $\Pi.\mathsf{Encap}$ (with other $t - z$ different attributes).

[11] Recall that the function $s(f)$ denotes the size of the extra part of the secret key, excluding the description of the function.

Hybrid H_t: This hybrid is almost same to H_{t-1}, except that $\mathcal{A}$ is given the following ciphertext

$$\mathsf{CT}_t := (\boldsymbol{r}, \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_1)), \ldots, \Pi.\mathsf{Encap}(\mathsf{mpk}, (\boldsymbol{x}, r_t))),$$

In this hybrid, it is clear that the ciphertext CT_t is generated as $\Pi_{\|}^{n',t}.\mathsf{Encap}^*$.

Then, it suffices to prove the computational indistinguishability between H_t and H_{t+1} for $z \in [t-1]$ □

Claim 6.5 *Suppose the valid/invalid ciphertext of the underlying* AB-wHPS *is selective or adaptive indistinguishability, then the above hybrids* H_z *and* H_{z+1} *are selective or adaptive indistinguishability for any* $z \in [t-1]$.

Proof We prove this claim through establishing a reduction from the valid/inva- lid ciphertext of the underlying AB-wHPS to the indistinguishability between H_z and H_{z+1}. This means if there is an efficient adversary $\mathcal{D}$ who can distinguish H_z from H_{z+1} with advantage ε, then we can construct an efficient reduction $\mathcal{B}$ to break the corresponding indistinguishability of underlying AB-wHPS with ε. Here, we just describe the reduction in the case of adaptive indistinguishability (underlying AB-wHPS), and note that a similar argument can be carried to the selective security in a straight-forward way.

Let $\mathcal{A}$ be the adversary for the ciphertext indistinguishability experiment for the amplified AB-wHPS, and $\mathcal{D}$ be a distinguisher that distinguishes H_z from H_{z+1} with a non-negligible advantage for some $z \in [t-1]$. Now we describe the reduction $\mathcal{B}$ that breaks the ciphertext indistinguishability of the underlying AB-wHPS when interacting with the challenger $\mathcal{C}$.

Setup $\mathcal{B}$ simulates either the hybrid H_z or H_{z+1} by running $\mathcal{A}$ in the following way.

1. $\mathcal{B}$ first get a master public-key mpk from the challenger $\mathcal{C}$ for the underlying AB-wHPS Π.
2. Then $\mathcal{B}$ forwards this mpk to the adversary $\mathcal{A}$ for the amplified AB-wHPS $\Pi_{\|}^{n',t}$.
3. At the same time, $\mathcal{B}$ sets a table $T = \emptyset$.

Test Stage 1 $\mathcal{B}$ answers the secret key queries of $\mathcal{A}$ in the following way.

1. $\mathcal{A}$ sends a function $f \in \mathcal{F}$ to $\mathcal{B}$ for a secret key query.
2. $\mathcal{B}$ first checks whether there exists an item containing this f in the table T.
   - If yes, $\mathcal{B}$ returns the corresponding secret key sk_f in T to $\mathcal{A}$.
   - Otherwise, $\mathcal{B}$ goes to the next step 3.
3. $\mathcal{B}$ sets $\hat{f}^i = \hat{f}^i_{f,h_i} \in \mathcal{F} \wedge_{\|} \mathcal{H}$ for every $i \xleftarrow{\$} [n']$,.
4. Then $\mathcal{B}$ sends all $\hat{f}^i$ to $\mathcal{C}$ to conduct secret key query for AB-wHPS, and thus get $\mathsf{sk}_{\hat{f}^i}$ as a respond.
5. Finally, $\mathcal{B}$ sends $\mathsf{sk}_f := \left(\mathsf{sk}_{\hat{f}^1}, \mathsf{sk}_{\hat{f}^2}, \ldots, \mathsf{sk}_{\hat{f}^{n'}}\right)$ as the secret key for f to $\mathcal{A}$, and stores the tuple (f, sk_f) as an item into the table T.

Challenge stage $\mathcal{B}$ simulates the challenge ciphertext to $\mathcal{A}$ as follows.

1. $\mathcal{A}$ choose any $\boldsymbol{x}^* \in \mathcal{X}$ satisfying that there is at most one function $f \in \mathcal{F}$ such that $f(x^*) = 1$ had been queried in Test Stage 1, as the challenge attribute to conduct the challenge query.
2. For a randomly chosen subset $\boldsymbol{r} := \{r_1, \ldots, r_t\} \subseteq [n']$, $\mathcal{B}$ sets attribute $\boldsymbol{x}^*_{z+1} = (\boldsymbol{x}^*, r_{z+1})$.

3. Then $\mathcal{B}$ send attribute $\boldsymbol{x}^*_{z+1}$ to $\mathcal{C}$ for the challenge query with respect to the underlying AB-wHPS.
4. Next, $\mathcal{B}$ obtains a ciphertext $\mathsf{CT}^*_{z+1} \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}(\boldsymbol{x}^*_{z+1})$ or $\mathsf{AB\text{-}wHPS.\,Encap}^*(\boldsymbol{x}^*_{z+1})$ depending on a random $b \in \{0, 1\}$ as the challenge ciphertexts from $\mathcal{C}$.
5. Furthermore, $\mathcal{B}$ sets $\boldsymbol{x}^*_i = (\boldsymbol{x}, r_i)$ for $i \in [t]$, and then calculates

$$\left\{\mathsf{CT}^*_i \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}^*(\boldsymbol{x}^*_i)\right\}_{i\in[z]}$$

and

$$\left\{\mathsf{CT}^*_i \xleftarrow{\$} \mathsf{AB\text{-}wHPS.Encap}(\boldsymbol{x}^*_i)\right\}_{i\in[t]\setminus[z+1]}$$

by himself.
6. $\mathcal{B}$ collects all ciphetexts CT^*_i for $i \in [t]$ together to construct $(\mathsf{CT}^*_1, \ldots, \mathsf{CT}^*_t)$ according to the indexes of these ciphertexts.
7. Finally, $\mathcal{B}$ sends this matrix $\mathsf{CT}^* := (\boldsymbol{r}, \mathsf{CT}^*_1, \ldots, \mathsf{CT}^*_t)$ as the challenge encapsulation ciphertext to $\mathcal{A}$.

Test stage 2 $\mathcal{B}$ answers the secret key queries of $\mathcal{A}$ as in Test Stage 1, but with a restriction that there is at most one function $f \in \mathcal{F}$ such that $f(x^*) = 1$ can been queried in Test Stage 1 and 2.

Output $\mathcal{B}$ simulates the output of the experiment and obtain a view H, which is either H_z or H_{z+1} as we will prove below. Finally, $\mathcal{B}$ outputs $\mathcal{D}(\mathsf{H})$.

Next, we analyze the advantage of $\mathcal{B}$. We observe that $\mathcal{B}$ perfectly simulates one of the two hybrids: if the challenge ciphertext from $\mathcal{C}$ is valid, then the amplified AB-wHPS challenge ciphertext CT^* is generated according to H_z, and otherwise H_{z+1}. Thus, the advantage of $\mathcal{B}$ is the same as that of $\mathcal{D}$ in distinguishing H_z from H_{z+1}, i.e., a non-negligible advantage ε. Thus, $\mathcal{B}$ breaks the ciphertext indistinguishability of the underlying AB-wHPS with advantage ε, which reaches a contradiction. This completes the proof of this claim. □

Lemma 6.4 follows directly from Claim 6.5 by a standard hybrid argument. □
In summary, we complete the proof of the first part of theorem. □

Combining Theorems 3.12 and 6.3, we obtain the following corollary.

Corollary 6.6 *Assume there exists an* ABE *scheme with the message space* $\mathbb{Z}_m$ *for the function class* $\mathcal{F} \wedge_{\|} \mathcal{H} \wedge_{\|} \mathcal{G}$*, where* $\mathcal{G}$ *with parameters* m, n *and* $\mathcal{H}$ *with parameter* n' *are as Definitions 3.9 and 6.1, then there exists an amplified* AB-wHPS *with the encapsulated-key-space* $\mathbb{Z}^t_m$ *for the function class* $\mathcal{F}$*.*

6.2 Locally computable extractor

Definition 6.7 (Locally computable extractor, [41, Definition 6]) An extractor $\mathsf{Ext} : \{0, 1\}^n \times \{0, 1\}^d \to \{0, 1\}^v$ is said to be t-locally computable if for every $r \in \{0, 1\}^d$, $\mathsf{Ext}(\boldsymbol{x}, r)$ depends only on t-bits of $\boldsymbol{x} \in \{0, 1\}^n$.

For our application (constructing leakage-resilient encryption in the BRM), we need a generalized variant of the above notion. Let $\boldsymbol{x} \in \{0, 1\}^{nk}$ be a vector. We can view it as a concatenation of n vectors $\boldsymbol{x}_i \in \{0, 1\}^k$ for $i \in [n]$, i.e., $\boldsymbol{x} = (\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_n^\top)^\top$. In this case, each $\boldsymbol{x}_i \in \{0, 1\}^k$ can be viewed as a symbol of some larger alphabet, i.e., $\Gamma = \{0, 1\}^k$, and we will need a locally computable extractor for Γ as follow.

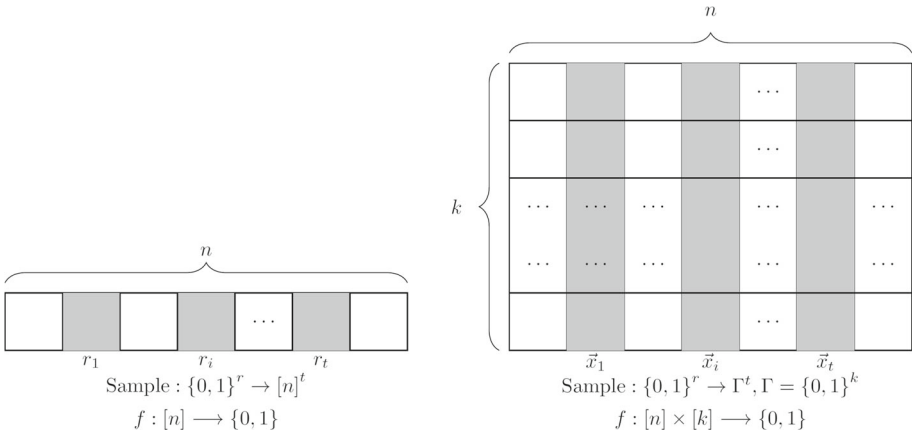

Fig. 1 Depiction of averaging samplers

Definition 6.8 (*Locally computable extractor for larger alphabets*) Let $\Gamma = \{0,1\}^k$ be some alphabet. An extractor $\mathsf{Ext} : \Gamma^n \times \{0,1\}^d \to \{0,1\}^v$ is t-locally computable with respect to Γ, if for every $\boldsymbol{r} \in \{0,1\}^d$, $\mathsf{Ext}(\boldsymbol{x}, \boldsymbol{r})$ depends only on t symbols of $\boldsymbol{x} = (\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_n^\top)^\top \in \Gamma^n$.

Generally, a locally computable extractor can be obtained in two steps [41]: (1) the extractor uses part of the seed to select t bits (or symbols) of $\boldsymbol{x}$, and (2) the remaining seed is used to apply a standard extractor on the selected bits/symbols in the previous step. Vadhan [41] showed that as long as the selection in step (1) achieves an averaging sampler, then the combined steps would achieve a locally computable extractor. Below, we summarize the result of Vadhan [41] below, and recall the formal notion of an averaging sampler.

Definition 6.9 (*Averaging sampler*, [41, Definition 8]) A function $\mathsf{Samp} : \{0,1\}^r \to [n]^t$ is a (μ, θ, γ) averaging sampler, if for every function $f : [n] \to \{0,1\}$ with average value $\frac{1}{n}\sum_i f(i) \geq \mu$,

$$\Pr_{(i_1,\ldots,i_t) \xleftarrow{\$} \mathsf{Samp}(U_r)} \left[\frac{1}{t} \sum_{j=1}^{t} f(i_j) < \mu - \theta \right] \leq \gamma.$$

In order to understand such an averaging sampler more clearly, we depict it in the left side of Fig. 1.

Next, we present a theorem by Vadhan in [41] that describes detailed requirements for a locally computable extractor.

Theorem 6.10 ([41, Theorem 10]) *Suppose that* $\mathsf{Samp} : \{0,1\}^r \to [n]^t$ *is an* (μ, θ, γ) *averaging sampler with distinct samples for* $\mu = (\delta - 2\tau)/\log(1/\tau)$ *and* $\theta = \tau/\log(1/\tau)$, *and* $\mathsf{Ext} : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}^v$ *is a strong* $((\delta - 3\tau)t, \varepsilon)$ *extractor. Define* $\mathsf{Ext}' : \{0,1\}^n \times \{0,1\}^{r+d} \to \{0,1\}^v$ *by*

$$\mathsf{Ext}'(\boldsymbol{x}, (\boldsymbol{y}_1, \boldsymbol{y}_2)) = \mathsf{Ext}(\boldsymbol{x}_{\mathsf{Samp}(\boldsymbol{y}_1)}, \boldsymbol{y}_2).$$

Then Ext' *is a* t*-local strong* $(\delta n, \varepsilon + \gamma + 2^{-\Omega(\tau n)})$ *extractor.*

As we mentioned above, our application needs a locally computable extractor for larger alphabets, which may not be implied directly from Theorem 6.10. To tackle this issue, we

define the following sampling procedure **Sampler 1** that outputs t distinct symbols of samples, and then prove that **Sampler 1** is in fact a good averaging sampler as needed in Theorem 6.10. This would imply a locally computable extractor for larger alphabets as required in our application.

Notations for the sampling Before describing the algorithm, we set up some notations as follows. Let $\Gamma = \{0,1\}^k$ and $\boldsymbol{x} = (\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_n^\top)^\top \in \Gamma^n$ be a vector of n symbols, where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ik})^\top \in \Gamma = \{0,1\}^k$ for $i \in [n]$. Let S denote a subset of $[n] \times [k]$, i.e. S contains tuples $(i, j) \in [n] \times [k]$ as its elements. In this case, we define $\boldsymbol{x}_S = \{x_{ij}\}_{(i,j)\in S}$. Then, we define **Sampler 1** as below. **Sampler 1:** Sample a random subset R of $[n]$ that contains t elements, i.e., $R = \{r_1, \ldots, r_t\}$, and output $S := \{(r_i, j)\}_{i\in[t], j\in[k]}$.

In order to understand such **Sampler 1** more clearly, we depict it in the right side of Fig. 1. Then we derive the following lemma.

Lemma 6.11 *For any $\kappa \in \mathbb{Z}$, $\mu, \theta \in (0,1]$ and $\gamma = 2\kappa \exp(-t\theta^2/4) + \left(\frac{t(t-1)}{2n}\right)^\kappa$, **Sampler 1** is a (μ, θ, γ) averaging sampler.*

Proof According to the natural bijection between $[nk]$ and $[n] \times [k]$, to prove that **Sampler 1** is a good averaging sampler as Definition 6.9, it suffices to show that for any $f : [n] \times [k] \to [0,1]$ such that $\frac{1}{nk}\sum_{i\in[n], j\in[k]} f(i,j) \geq \mu$, the following inequality holds:

$$\Pr_{S \xleftarrow{\$} \textbf{Sampler 1}} \left[\frac{1}{|S|} \sum_{(i,j)\in S} f(i,j) < \mu - \theta \right] \leq \gamma. \tag{1}$$

In order to do this, our first transfer the algorithm **Sampler 1** into the other statistically close algorithm **Sampler 2** (via Claim 6.12), and then prove that the above inequality (1) holds for **Sampler 2** through using a Chernoff bound argument (via Claim 6.13). Thus, we conclude that **Sampler 1** is a good averaging sampler with overwhelming probability. Furthermore, we conclude that **Sampler 1** with any strong extractor yields a locally computable extractor for larger alphabets.

Particularly, we define **Sampler 2** as follows.

Sampler 2

1. Sample $R = \{r_1, \ldots, r_t\}$ from $[n]^t$ uniformly at random.
   - If all elements are distinct, then output $S := \{(r_i, j)\}_{i\in[t], j\in[k]}$ and terminate.
2. Otherwise, i.e., there is a repeated element, discard the whole sample and redo Step 1. Note: the algorithm will only redo Step 1 up to κ times. If the algorithm does not produce an output by then, then output $\perp$.

Next we analyze **Sampler 1** and **Sampler 2** by the following two claims. □

Claim 6.12 *For a set X consisting of $n = n(\kappa)$ different blocks and the parameters $t = t(\kappa)$ such that $1 \leq t \leq n$, the output distributions of Sample 1 and Sample 2 are statistically close.*

Proof We notice that the distribution of Sampler 1 is identical to that of Sampler 2 conditioned on non-$\perp$ values. Therefore, their statistical distance is bounded by the probability that Sampler 2 does not terminate in κ steps. Let T denote the event that Sampler 2 selects distinct elements at a particular round (and thus terminates). We have

$$\Pr[T] = \frac{n(n-1)\cdots(n-t+1)}{n^t}.$$

Since every round of Sample 2 is independent of others, we know the probability of Sample 2 outputs $\perp$ is

$$(1 - \Pr[T])^{\kappa} = \left(1 - \frac{n(n-1)\cdots(n-t+1)}{n^t}\right)^{\kappa} \leq \left(1 - \frac{(n-t+1)^t}{n^t}\right)^{\kappa}.$$

Therefore, for $1 \leq t \leq n$, it holds that $0 < \frac{(n-t+1)^t}{n^t} \leq 1$. the statistical distance between two output distributions is at most $\left(1 - \frac{(n-t+1)^t}{n^t}\right)^{\kappa} \leq \mathsf{negl}(\kappa)$. □

Claim 6.13 *For any μ, t, θ, n, Sampler 2 is a (μ, θ, γ) average sampler conditioned on non-$\perp$ output, where $\gamma = 2\kappa \exp(-t\theta^2/4)$.*

Proof As we discussed above, it suffices to show that for any $f : [n] \times [k] \to \{0, 1\}$ such that $\frac{1}{nk}\sum_{i\in[n], j\in[k]} f(i, j) \geq \mu$, we have:

$$\Pr_{S \xleftarrow{\$} \mathbf{Sampler2}}\left[\frac{1}{|S|}\sum_{(i,j)\in S} f(i, j) < \mu - \theta\right] \leq \gamma,$$

conditioned on $S \neq \perp$.

In particular, let $f : [n] \times [k] \to \{0, 1\}$ be a function such that $\mu_f := \frac{1}{nk}\sum_{i\in[n], j\in[k]} f(i, j) \geq \mu$. Let $r_1, \ldots, r_t$ be i.i.d. random variables sampled from $[n]$, and $S_i = \{(r_i, j)\}_{j\in[k]}$. Clearly, $S_1, \ldots, S_t$ are the choices of Sampler 2 at a particular round, and they are also i.i.d. random variables. If $r_1, \ldots, r_t$ are distinct, then Sampler 2 will output $S = \{S_1, \ldots, S_t\}$. Next we denote random variables $\mu_{r_i} := \frac{1}{k}\sum_{j=1}^{k} f(r_i, j)$ for $i \in [t]$, and clearly, μ_{r_i}'s are also i.i.d. random variables with the same expectation

$$E[\mu_{r_i}] = \sum_{i'\in[n]} \frac{1}{k}\sum_{j\in[k]} f(r_i, j)\mathsf{Pr}[r_i = i'] = \frac{1}{nk}\sum_{i\in[n], j\in[k]} f(i, j) = \mu_f.$$

Therefore, by the Chernoff bound, we have:

$$\Pr\left[\left|\frac{1}{t}\sum_{i=1}^{t}\mu_{r_i} - \mu_f\right| \geq \theta\right] \leq 2\exp(-t\theta^2/4),$$

for any $\theta, t > 0$. As $\mu_f \geq \mu$ from the assumption. Thus for this particular round, we have

$$\begin{aligned}\Pr\left[\frac{1}{t}\sum_{i=1}^{t}\mu_{r_i} \leq \mu - \theta\right] &\leq \Pr\left[\frac{1}{t}\sum_{i=1}^{t}\mu_{r_i} \leq \mu_f - \theta\right] \\ &\leq \Pr\left[\left|\frac{1}{t}\sum_{i=1}^{t}\mu_{r_i} - \mu_f\right| \geq \theta\right] \\ &\leq 2\exp(-t\theta^2/4).\end{aligned} \tag{2}$$

Then by a union bound over all rounds, we have:

$$\begin{aligned}
&\Pr_{S \xleftarrow{\$} \textbf{Sampler2}}\left[\frac{1}{|S|}\sum_{(i,j)\in S} f(i,j) < \mu - \theta\right] \\
&\leq \Pr\left[\exists \text{ a round such that } \frac{1}{t}\sum_{i=1}^{t}\mu_{r_i} \leq \mu - \theta\right] \\
&\leq 2\kappa \exp(-t\theta^2/4).
\end{aligned}$$

This concludes the proof of the claim. □

The proof of the lemma follows by the above Claims 6.12 and 6.13. □

Furthermore, by applying the **Sample 1** to Theorem 6.10 with the following parameters setting, we derive the following theorem.

Parameter setting Taking κ as the security parameter, we set all the parameters in the following way: $k = \mathsf{poly}(\kappa), n = \mathsf{poly}(\kappa), t = \kappa \log^3(nk), \delta = \frac{1}{\log(nk)}, \tau = \frac{1}{6\log(nk)}, \mu = \frac{2}{3\log(nk)\log(6\log(nk))}, \theta = \frac{1}{6\log(nk)\log(6\log(nk))}, \gamma = 2\kappa\exp(-t\theta^2/4) + \left(\frac{t(t-1)}{2n}\right)^{\kappa}, \varepsilon = \mathsf{negl}(\kappa)$.

Theorem 6.14 *Let* $\Gamma = \{0,1\}^k$, $\mathsf{Samp} : \{0,1\}^r \to [n]^t$ *be the* ***Sampler 1*** *(as a* (μ, θ, γ) *average sampler), and let* $\mathsf{Ext} : \Gamma^t \times \{0,1\}^d \to \{0,1\}^v$ *be a strong* $((\delta - 3\tau)tk, \varepsilon)$ *extractor. Define* $\mathsf{Ext}' : \Gamma^n \times \{0,1\}^{r+d} \to \{0,1\}^v$ *as*

$$\mathsf{Ext}'(\boldsymbol{x}, (\boldsymbol{y}_1, \boldsymbol{y}_2)) = \mathsf{Ext}(\boldsymbol{x}_{\mathsf{Samp}(\boldsymbol{y}_1)}, \boldsymbol{y}_2).$$

Then Ext' *is a* t*-block-local strong* $(\delta nk, \varepsilon + \gamma + 2^{-\Omega(\tau n)})$ *extractor, where* $\varepsilon + \gamma + 2^{-\Omega(\tau n)} = \mathsf{negl}(\kappa)$ *according to the setting of parameters.*

6.3 Leakage-resilient encryption in the bounded-retrieval model

In this section, we construct leakage-resilient encryption schemes in the BRM, through combining an random extractor with an amplified AB-wHPS presented in Sect. 6.1. Below, we give the specific construction of leakage resilient ABE scheme in the BRM from an amplified AB-wHPS.

Construction 6.15 (Construction in the BRM) *Let* $\Pi = \mathsf{AB\text{-}wHPS}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Encap}^*, \mathsf{Decap}\}$ *be an amplified* AB-wHPS *with integer parameters* n', t*, the encapsulated-key-space* $\mathcal{K}^t$ *and attribute space* $\mathcal{X} = \{0,1\}^*$ *for a class of policy functions* $\mathcal{F} = \{f : \{0,1\}^* \to \{0,1\}\}$*. Let* $\mathsf{Ext} : \mathcal{K}^t \times \mathcal{S} \to \mathcal{M}$ *be a strong extractor, where three sets* $\mathcal{K}, \mathcal{S}, \mathcal{M}$ *are efficient ensembles,* k *denotes the size of* $\mathcal{K}$*. Furthermore, assume that* $\mathcal{M}$ *is an additive group. Then, an* ABE *scheme* $\Pi_{\mathcal{F}} = \Pi_{\mathcal{F}}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *with message space* $\mathcal{M}$ *and policy function class* $\mathcal{F}$ *can be constructed as follows:*

- $\Pi_{\mathcal{F}}.\mathsf{Setup}(1^\kappa)$*: The algorithm runs* $(\mathsf{mpk}^{\Pi}, \mathsf{msk}^{\Pi}) \xleftarrow{\$} \Pi.\mathsf{Setup}(1^\kappa)$*, and outputs* $\mathsf{mpk} := \mathsf{mpk}^{\Pi}$*, and* $\mathsf{msk} := \mathsf{msk}^{\Pi}$.
- $\Pi_{\mathcal{F}}.\mathsf{KeyGen}(\mathsf{msk}, f)$*:* $\Pi_{\mathcal{F}}.\mathsf{KeyGen}(\mathsf{msk}, f)$*: Given a master secret-key* msk *and a function* $f \in \mathcal{F}$ *as input, the algorithm runs*
  $\mathsf{sk}_f^{\Pi} \xleftarrow{\$} \mathsf{AB\text{-}wHPS.KeyGen}(\mathsf{msk}, f)$ *and output* $\mathsf{sk}_f := \mathsf{sk}_f^{\Pi}$.

– $\Pi_{\mathcal{F}}.\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x}, \mu)$*: Given a master public-key* mpk*, an attribute* $\boldsymbol{x} \in \{0,1\}^*$ *and a message* $\mu \in \mathcal{M}$ *as input, the algorithm runs* AB-wHPS.Encap *to generate* $(CT', \boldsymbol{k}) \leftarrow \mathsf{AB\text{-}wHPS.Encap}(\mathsf{mpk}, \boldsymbol{x})$ *with* $\boldsymbol{k} \in \mathcal{K}^t$*, and then samples* $s \xleftarrow{\$} \mathcal{S}$*. Furthermore, the algorithm computes and outputs*

$$ct = (s, ct_0, ct_1) = (s, CT', \mu + \mathsf{Ext}(\boldsymbol{k}, s)).$$

– $\Pi_{\mathcal{F}}.\mathsf{Dec}(\mathsf{sk}_f, ct)$*: Given a ciphertext* $ct = (s, ct_0, ct_1)$ *and a secret key* sk_f *as input, the algorithm runs* AB-wHPS.Decap *to generate* $\boldsymbol{k} = \mathsf{AB\text{-}wHPS.Decap}(\mathsf{sk}_f, ct_0)$ *with* $\boldsymbol{k} \in \mathcal{K}^t$*, and then output* $\mu = ct_1 - \mathsf{Ext}(\boldsymbol{k}, s)$.

Parameter setting For security parameter κ, we set the system parameters as follows: $k = \mathsf{poly}(\kappa)$, $n' = \mathsf{poly}(\kappa)$, $t = \kappa \log^3(n'k)$, $\delta = \frac{1}{\log(n'k)}$, $\tau = \frac{1}{6\log(n'k)}$, $\varepsilon = \mathsf{negl}(\kappa)$. Moreover, for the proof of leakage-resilience in the BRM, we let $\mathsf{Ext} : \mathcal{K}^t \times \mathcal{S} \to \mathcal{M}$ be a $((\delta - 3\tau)tk, \varepsilon)$-extractor.

Next, we prove that the construction is a leakage resilient ABE in the BRM. Our proof uses a technique of locally computable extractors [41], i.e., Theorem 6.14, in a black-box way.

Theorem 6.16 *Assume* Π *is a selectively (or adaptively, resp.) secure amplified* AB-wHPS *with integer parameters* $n', t = \kappa \log^3(n'k)$ *for the policy function class* $\mathcal{F}$*, and* $\mathsf{Ext} : \mathcal{K}^t \times \mathcal{S} \to \mathcal{M}$ *be a strong extractor. Then the above* ABE *scheme* $\Pi_{\mathcal{F}} = \Pi_{\mathcal{F}}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *for* $\mathcal{F}$ *is a selectively (or adaptively, resp.)* ℓ*-leakage-resilient attribute-based encryption scheme with message space* $\mathcal{M}$ *in the* BRM *where* $\ell = kn' - \frac{kn'}{\log(kn')}$.

Particularly, $\Pi_{\mathcal{F}}$ *is also*

– *an* ℓ*-leakage-resilient public-key encryption scheme in the* BRM *with* $\ell = kn' - \frac{kn'}{\log(kn')}$*, if* $\mathcal{F}$ *contains only a single function that always outputs* 1.
– *a selectively (or adaptively, resp.)* ℓ*-leakage-resilient identity-based encryption scheme in the* BRM *with* $\ell = kn' - \frac{kn'}{\log(kn')}$*, if* $\mathcal{F}$ *contains the following comparison functions, i.e., each function* $f_{\boldsymbol{y}} \in \mathcal{F}$ *is indexed by a vector* $\boldsymbol{y}$*, and* $f_{\boldsymbol{y}}(\boldsymbol{x}) = 1$ *if and only if* $\boldsymbol{y} = \boldsymbol{x}$.

Moreover,

1. *Public-key (resp. master public-key) size of* $\Pi_{\mathcal{F}}$ *is the same as that of* Π*, which is not dependent on leakage parameter* ℓ.
2. *The locality-parameter is* $t = \kappa \log^3(n'k)$*. Thus, the size of secret-key accessed during decryption depends on* t*, but not* ℓ.
3. *The ciphertext-size/encryption-time/decryption-time of* $\Pi_{\mathcal{F}}$ *depends on* t*, but not* ℓ.

Proof Similar to the proof for leakage-resilience in the relative model, we just prove the general case of ABE for general functions $\mathcal{F}$ in the BRM. Then, the results for IBE and PKE can be proved similarly, since IBE and PKE are special cases of ABE for equation-testing functions and constant function, respectively. The correctness of this ABE scheme $\Pi_{\mathcal{F}}$ follows naturally from that of amplified AB-wHPS Π. Below we focus on proving leakage resilience.

Let us denote $\boldsymbol{r} \in \{0,1\}^*$ as the randomness used to sample random subset $\{r_1, \ldots, r_t\} \subseteq [m]$ in the construction of amplified AB-wHPS, i.e., $\boldsymbol{r} = (r_1, \ldots, r_t)^\top$. That is, for $\boldsymbol{k}' = (k_1, \ldots, k_{n'})^\top \in \mathcal{K}^{n'}$, there exists a random sampling algorithm $\mathsf{Samp}_{\boldsymbol{r}}(\boldsymbol{k}')$ that samples a random subset $\{r_1, \ldots, r_t\} \subseteq [m]$ and outputs $\boldsymbol{k} = (k_{r_1}, \ldots, k_{r_t})^\top$. Similarly, for $(\mathsf{CT}_1, \ldots, \mathsf{CT}_{n'}) \in \mathcal{CT}^{n'}$, $\mathsf{Samp}_{\boldsymbol{r}}(\mathsf{CT}_1, \ldots, \mathsf{CT}_{n'})$ outputs $(\mathsf{CT}_{r_1}, \ldots, \mathsf{CT}_{r_t})$.

We define $\mathsf{Ext}' : \mathcal{K}^{n'} \times (\{0,1\}^* \times \mathcal{S}) \to \mathcal{M}$ by

$$\mathsf{Ext}'(\boldsymbol{k}', \boldsymbol{r}, s) = \mathsf{Ext}(\boldsymbol{k}_{\mathsf{Samp}_{\boldsymbol{r}}(\boldsymbol{k}')}, s).$$

As a result, the ciphertext CT for $\Pi_{\mathcal{F}}$ can be rewritten as

$$\mathsf{ct} = (\boldsymbol{r}, s, \mathsf{CT}_{r_1}, \ldots, \mathsf{CT}_{r_t}, m + \mathsf{Ext}'(\boldsymbol{k}', \boldsymbol{r}, s)).$$

From Theorem 6.14 and the setting of parameters for Construction 6.15, we can conclude that $\mathsf{Ext}' : \mathcal{K}^{n'} \times (\{0,1\}^* \times \mathcal{S}) \to \mathcal{M}$ is a t-locally computable strong $(\frac{n'k}{\log(n'k)}, \varepsilon + \gamma + 2^{-\Omega(\tau n' k)})$ extractor for alphabets $\mathcal{K}$. Thus, the leakage resilience of $\Pi_{\mathcal{F}}$ can be proved through a sequence of hybrids similar to the proof of Theorem 5.2 in the relative leakage model.

The allowed leakage length is $kn' - \frac{kn'}{\log(kn')}$. At the same time, it is clear that all efficiency parameters of $\Pi_{\mathcal{F}}$ are not dependent on leakage parameter ℓ. Thus, $\Pi_{\mathcal{F}}$ is a $\left(kn' - \frac{kn'}{\log(kn')}\right)$-leakage resilient ABE in the BRM. □

Combining Corollary 6.6 and Theorem 6.16, we obtain the following results. Assume there exists an ABE scheme with the message space $\mathbb{Z}_m$ for the function class $\mathcal{F} \wedge_{\|} \mathcal{H} \wedge_{\|} \mathcal{G}$, where $\mathcal{G}$ with parameters m, n and $\mathcal{H}$ with parameter n' are as defined in Definitions 3.9 and 6.1, and the key-length (of the extra part, excluding the function description of f) of this underlying ABE scheme for policy function f is $s(f)$. Then the largest allowed leakage length of the above ABE (or IBE or PKE) scheme $\Pi_{\mathcal{F}}$ for the function class $\mathcal{F}$ is $\ell = (kn' - \frac{kn'}{\log(kn')})$ with $k = n \log m$ and the key-length of $\Pi_{\mathcal{F}}$ for f is $|\mathsf{sk}_f| = n'(n \log m + \log n' + |f| + s(\hat{f}_{f,h,g_{\boldsymbol{y}}}))$.

Furthermore, if the secret key size $s(\hat{f}_{f,h,g_{\boldsymbol{y}}}))$is succinct, i.e., $s(\hat{f}_{f,h,g_{\boldsymbol{y}}}) = o(|\hat{f}_{f,h,g_{\boldsymbol{y}}}|) = o(n \log m + \log n' + |f|)$, then we can set sufficiently large n, m, n' such that $(\log n' + |f|) = o(n \log m)$. Consequently, the leakage rate of this scheme $\Pi_{\mathcal{F}}$ is $\frac{kn' - \frac{kn'}{\log(kn')}}{n'(n \log m + \log n' + |f| + s(\hat{f}_{f,h,g_{\boldsymbol{y}}}))}$ $= \frac{1 - \frac{1}{\log(nn' \log m)}}{1 + \frac{\log n' + |f| + s(\hat{f}_{f,h,g_{\boldsymbol{y}}})}{n \log m}} \approx 1 - o(1)$, achieving the desired optimal leakage rate.

Finally, by combining Corollary 4.8 and Theorem 6.16, we obtain the following Corollary.

Corollary 6.17 *Assuming* LWE, *for all polynomial* $S = \mathsf{poly}(\kappa)$, *there exist* $1 - o(1)$ *leakage resilient* ABE *schemes in the* BRM, *which are*

1. *adaptively secure for the comparison functions;*
2. *adaptively secure for* t-CNF* *functions of size up to* S*;*
3. *selectively secure for general circuits of size up to* S.

For unbounded polynomial S, our schemes are still leakage resilient with the optimal rate for a smaller function class. See Remark 5.4 for the discussion.

7 Extension II: leakage on multiple keys

Our prior ABE constructions from AB-wHPS only achieve leakage resilience in the one-key setting where the adversary can only leak on one of the all possible decrypting keys with respect to the challenge attribute. In this section, we show how to achieve leakage resilience in the *multiple-key* setting where the attacker can obtain leakage on ω possible decrypting keys for any bounded polynomial ω. Our construction leverages the normal AB-wHPS (where the ciphertext indistinguishability holds when the adversary gets one decrypting key) and a threshold secret sharing scheme, following the bootstrapping idea of the work [24].

Below, we first introduce a useful lemma which is the key principle for the following parameter setting. Notice that this lemma has been previously given as Lemma C.1 in [24]. However, it seems that their proof has certain flaws. Here, we prove it again in a much more formal way.

Lemma 7.1 *Let $\Gamma_1, \ldots, \Gamma_\omega$ be randomly chosen subsets of size $t+1$. Let $t_0 = \Theta(\omega^2 t \kappa^{\frac{1}{c}})$, and $n = \Theta(\omega^2 t)$. It holds*

$$\Pr\left[\left|\bigcup_{i\neq j}(\Gamma_i \cap \Gamma_j)\right| \leq t_0\right] = 1 - e^{-\Omega(\kappa)},$$

where the probability is over the random choice of the subsets $\Gamma_1, \ldots, \Gamma_\omega$.

Proof For all $i, j \in [\omega]$ such that $i \neq j$, we use X_{ij} to denote a random variable, which represents the size of the intersection of Γ_i and Γ_i. Then, we define the following random variable

$$X = \sum_{i,j\in[\omega], i\neq j} X_{ij}.$$

Clearly, it holds $\left|\bigcup_{i\neq j}(\Gamma_i \cap \Gamma_j)\right| \leq X$. Thus, for the proof of this lemma, it is sufficient to get a meaningful upper bound for X.

Notice also that for a fixed set Γ_i and a randomly chosen set Γ_j, X_{ij} follows a hypergeometric distribution, where $t+1$ serves as the number of success states and number of trials, and n is the population size. In this case, for $0 < \delta < \frac{(t+1)^2}{n}$, there is an tail bound:

$$\Pr\left[X_{ij} \geq \frac{(t+1)^2}{n} + \delta(t+1)\right] \leq e^{-2\delta^2(t+1)}.$$

Furthermore, it holds

$$\begin{aligned}
&\Pr\left[X \geq \frac{\omega(\omega-1)}{2}\left(\frac{(t+1)^2}{n} + \delta(t+1)\right)\right] \\
&= \Pr\left[\sum_{i,j\in[\omega], i\neq j} X_{ij} \geq \frac{\omega(\omega-1)}{2}\left(\frac{(t+1)^2}{n} + \delta(t+1)\right)\right] \\
&\leq \Pr\left[\bigcup_{i\neq j}\left(X_{ij} \geq \frac{(t+1)^2}{n} + \delta(t+1)\right)\right] \\
&\leq \frac{\omega(\omega-1)}{2}\Pr\left[X_{ij} \geq \frac{(t+1)^2}{n} + \delta(t+1)\right] \\
&\leq \frac{\omega(\omega-1)}{2} e^{-2\delta^2(t+1)}.
\end{aligned}$$

Thus, setting $n = \Theta(\omega^2 t)$, $t_0 = \Theta(\omega^2 t \kappa^{\frac{1}{c}})$ for any constant c, we have

$$\Pr[X \geq t_0] \leq e^{-\Omega(\kappa)}.$$

□

Construction 7.2 (Extended leakage resilient ABE) *Let* $\Pi = \Pi.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Encap}^*, \mathsf{Decap}\}$ *be a* $(\log|\mathcal{K}|, \log|\mathcal{K}|)$*-universal* AB-wHPS *with the encapsulated-key-space* $\mathcal{K}$ *and attribute space* $\mathcal{X} = \{0,1\}^*$ *for a class of policy functions* $\mathcal{F} = \{f : \{0,1\}^* \to \{0,1\}\}$. *Let* $\mathsf{Ext} : \mathcal{K} \times \mathcal{S} \to \mathcal{M}$ *be a* $(\log|\mathcal{K}| - \ell, \varepsilon)$*-extractor, where* $\mathcal{K}, \mathcal{S}, \mathcal{M}$ *are efficient ensembles,* $\ell = \ell(\kappa)$ *is some parameter and* $\varepsilon = \varepsilon(\kappa) = \mathsf{negl}(\kappa)$ *is negligible. In addition, let* $(\mathsf{Share}, \mathsf{Rec})$ *be a* $(\hat{t}+1)$*-out-of-n threshold secret sharing scheme with respect to secret domain* $\mathcal{M}$*, an additive group.*

Then, a leakage-resilient ABE *scheme* $\Pi_{\mathcal{F}} = \Pi_{\mathcal{F}}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *with message space* $\mathcal{M}$ *for policy function class* $\mathcal{F}$ *can be constructed as follows:*

- $\Pi_{\mathcal{F}}.\mathsf{Setup}(1^\kappa, n)$*: The algorithm runs* $(\mathsf{mpk}_i^{\Pi}, \mathsf{msk}_i^{\Pi}) \xleftarrow{\$} \Pi.\mathsf{Setup}(1^\kappa)$ *for every* $i \in [n]$*, and outputs* $\mathsf{mpk} := \{\mathsf{mpk}_i^{\Pi}\}_{i\in[n]}$ *and* $\mathsf{msk} := \{\mathsf{msk}_i^{\Pi}\}_{i\in[n]}$.
- $\Pi_{\mathcal{F}}.\mathsf{KeyGen}(\mathsf{msk}, f)$*: Given a master secret-key* $\mathsf{msk} := \{\mathsf{msk}_i^{\Pi}\}_{i\in[n]}$ *and a function* $f \in \mathcal{F}$ *as input, the algorithm first chooses a random subset of cardinality* $\hat{t}+1$*, i.e.,* $\Gamma = \{r_1, \ldots, r_{\hat{t}+1}\} \subseteq [n]$*, and then runs* $\mathsf{sk}_f^{(r_i)} \xleftarrow{\$} \Pi.\mathsf{KeyGen}(\mathsf{msk}_{r_i}^{\Pi}, f)$ *for* $i \in [\hat{t}+1]$. *Finally, the algorithm outputs*

$$\mathsf{sk}_f := (\Gamma, \mathsf{sk}_f^{(r_1)}, \ldots, \mathsf{sk}_f^{(r_{\hat{t}+1})}).$$

- $\Pi_{\mathcal{F}}.\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x}, \mu)$*: Given a master public-key* $\mathsf{mpk} := \{\mathsf{mpk}_i^{\Pi}\}_{i\in[n]}$*, an attribute* $\boldsymbol{x} \in \mathcal{X} = \{0,1\}^*$ *and a message* $\mu \in \mathcal{M}$ *as input, the algorithm first runs* $(\mu_1, \ldots, \mu_n) \xleftarrow{\$} \mathsf{Share}(\mu)$. *Furthermore, the algorithm runs* $\Pi.\mathsf{Encap}$ *to generate* $(CT_i, k_i) \xleftarrow{\$} \Pi.\mathsf{Encap}(\mathsf{mpk}_i, \boldsymbol{x})$ *for every* $i \in [n]$. *Next, the algorithm samples* $s_1, \ldots, s_n \xleftarrow{\$} \mathcal{S}$*, and outputs*

$$\begin{aligned} ct &= (s_1, \ldots, s_n, ct_1, \ldots, ct_n, ct_{n+1}, \ldots, ct_{2n}) \\ &= (s_1, \ldots, s_n, CT_1, \ldots, CT_n, \mu_1 + \mathsf{Ext}(k_1, s_1), \ldots, \mu_n + \mathsf{Ext}(k_n, s_n)). \end{aligned}$$

- $\Pi_{\mathcal{F}}.\mathsf{Dec}(\mathsf{sk}_f, ct)$*: Given a ciphertext* $ct = (\{s_i\}_{i\in[n]}, \{ct_i\}_{i\in[2n]})$ *and a secret key* $\mathsf{sk}_f = (\Gamma, \{\mathsf{sk}_f^{(r_i)}\}_{i\in[\hat{t}+1]})$ *as input, the algorithm first runs* $\Pi.\mathsf{Decap}$ *to generate* $k_{r_i} = \Pi.\mathsf{Decap}(\mathsf{sk}_f^{(r_i)}, ct_{r_i})$ *and* $\mu_{r_i} = ct_{n+r_i} - \mathsf{Ext}(k_{r_i}, s_{r_i})$ *for every* $i \in [\hat{t}+1]$. *Then, the algorithm outputs* $\mu = \mathsf{Rec}(\mu_{r_1}, \ldots, \mu_{r_{\hat{t}+1}})$.

Parameter setting For security parameter κ, given any $\omega = \mathsf{poly}(\kappa)$, we set $\hat{t} = \Theta(\omega^2\kappa)$ and $n = \Theta(\omega^2\hat{t})$. For details, we refer readers to Lemma 7.1.

Our construction achieves a leakage resilient ABE in the multiple key setting. We summarize the results in the following theorem.

Theorem 7.3 *Assume* Π *is a selectively (or adaptively, resp.) secure* $(\log|\mathcal{K}|, \log|\mathcal{K}|)$*-universal* AB-wHPS *for the policy function class* $\mathcal{F}$*, and* $\mathsf{Ext} : \mathcal{K} \times \mathcal{S} \to \mathcal{M}$ *be a* $(\log|\mathcal{K}| - \ell, \mathsf{negl}(\kappa))$*-extractor. Then the above* ABE *scheme* $\Pi_{\mathcal{F}} = \Pi_{\mathcal{F}}.\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *for* $\mathcal{F}$ *is a selectively (or adaptively, resp.)* $(\ell(\kappa), \omega(\kappa))$*-leakage resilient attribute-based encryption scheme for* $\mathcal{F}$ *in the relative-leakage model, for any fixed bounded polynomial* $\omega(\kappa) = \mathsf{poly}(\kappa)$.

The corresponding leakage rate is $\frac{\ell(\kappa)}{(\hat{t}+1)(|\mathsf{sk}_f|+\log n)}$. *Furthermore, when the underlying secret keys* $(\mathsf{sk}_f^{(r_1)}, \ldots, \mathsf{sk}_f^{(r_{\hat{t}+1})})$ *form a block source under each leakage function, the corresponding leakage rate is* $\frac{\ell(\kappa)}{(|\mathsf{sk}_f|+\log n)}$.

Proof Clearly, the correctness of this ABE scheme $\Pi_{\mathcal{F}}$ follows naturally from that of AB-wHPS Π and $(t+1)$-out-of-n threshold secret sharing scheme (Share, Rec). Furthermore, the security of this ABE scheme can be argued through using a sequence of hybrids as follows.

Hybrid $\mathbf{H_0}$: This hybrid is defined to be the security experiment with (ℓ, ω)-leakage in Definition 2.11. In this hybrid, the view of $\mathcal{A}$ consists of the master public-key mpk, leakage information $\{h_i(\mathsf{sk}_{f_i})\}_{i\in[w]}$, and challenge ciphertext $\mathsf{ct} = (\{s_i\}_{i\in[n]}, \{\mathsf{ct}_i\}_{i\in[2n]})$, where $\mathsf{mpk} := \{\mathsf{mpk}_i^{\Pi}\}_{i\in[n]}$, $\mathsf{sk}_{f_i} := (\Gamma_i, \{\mathsf{sk}_{f_i}^{(r_{i,j})}\}_{j\in[t+1]})$ with $\Gamma_i = \{r_{i,1}, \ldots, r_{i,t+1}\} \subseteq [n]$, $f_i(\boldsymbol{x}^*) = 1$ and $i \in [\omega]$, $s_i \xleftarrow{\$} \mathcal{S}$ with $i \in [n]$, and

$$(\mathsf{ct}_i, k_i) \leftarrow \Pi.\mathsf{Encap}(\mathsf{mpk}_i, \boldsymbol{x}^*), \quad \mathsf{ct}_{n+i} = \mu_{b,i} + \mathsf{Ext}(k_i, s_i)$$

with $i \in [n]$ and $(\mu_{b,1}, \ldots, \mu_{b,t+1}) \xleftarrow{\$} \mathsf{Share}(\mu_b)$. Notice that the block leakage function $h_i : \{0,1\}^* \to \{0,1\}^{\ell}$ is chosen adaptively by the adversary before the challenge stage. Here, in the leakage query stage, $\mathcal{A}$ is allowed to query ω policy functions f_i's such that $f_i(\boldsymbol{x}^*) = 1$ with each $i \in [\omega]$. Recall that $\boldsymbol{x}^*$ is the challenge attribute.

Hybrid $\mathbf{H_1}$ This hybrid is almost identical to the H_0, except that for positive integer ω, the challenger chooses the random subsets $\Gamma_i = \{r_{i,1}, \ldots, r_{i,t+1}\} \subseteq [n]$ with each $i \in [\omega]$ in advance, and put them as parts of the master secret key, i.e., $\mathsf{msk} := (\{\mathsf{msk}_i^{\Pi}\}_{i\in[n]}, \{\Gamma_i\}_{i\in[\omega]})$. When the adversary requests the leakage queries on the challenge secret keys sk_{f_i} for $i \in [w]$, the challenger directly uses the pre-selected subset Γ_i to respond. Clearly, H_0 to H_1 are identical from the view of the adversary.

Hybrid $\mathbf{H_2}$ This hybrid is almost identical to the H_1, except the challenge ciphertext is computed in the following way:

Given the subsets $\Gamma_i = \{r_{i,j}\}_{j\in[t+1]}$ for $i \in [\omega]$, the challenger computes the union of Γ_i for $i \in [\omega]$, i.e., $\bar{\Gamma} = \bigcup_{i\in[n]} \Gamma_i \subseteq [n]$, and then partitions $[n]$ into two disjoint sets $\bar{\Gamma}$ and $[n] \setminus \bar{\Gamma}$. Then for each $r_{i,j} \in \bar{\Gamma}$, the challenger computes

$$\begin{aligned}&(\mathsf{ct}_{r_{i,j}}, k_{r_{i,j}}) \leftarrow \Pi.\mathsf{Encap}(\mathsf{mpk}_{r_{i,j}}, \boldsymbol{x}^*), \quad k'_{r_{i,j}} = \Pi.\mathsf{Decap}(\mathsf{sk}_{f_i}^{(r_{i,j})}, \mathsf{ct}_{r_{i,j}}),\\ &\mathsf{ct}_{n+r_{i,j}} = \mu_{b,r_{i,j}} + \mathsf{Ext}(k'_{r_{i,j}}, s).\end{aligned}$$

For other indices $r_{i,j} \in [n] \setminus \bar{\Gamma}$, the ciphertexts are computed in the same way as that of $\bar{\Gamma}$. Therefore, the only difference between H_0 and H_1 is the usage of $k_{r_{i,j}}$ and $k'_{r_{i,j}}$ in the computation of $\mathsf{ct}_{n+r_{i,j}}$ for all $r_{i,j} \in [n]$. In fact, $k_{r_{i,j}} = k'_{r_{i,j}}$ according to the correctness of the underlying AB-wHPS Π. Hence, H_1 and H_2 are identical.

Hybrid $\mathbf{H_3}$ This hybrid is almost same to H_2, except the challenge ciphertext is computed in the following way:

The challenger first computes the subset Γ_0 containing all elements $r_{i,j}$ that are included in more than one subset Γ_i for $i \in [\omega]$, such that $\Gamma_0 \subseteq \bar{\Gamma} \subseteq [n]$. Then for each $r_{i,j} \in [n] \setminus \Gamma_0 = ([n] \setminus \bar{\Gamma}) \cup (\bar{\Gamma} \setminus \Gamma_0)$, the challenger computes

$$\begin{aligned}&\mathsf{ct}'_{r_{i,j}} \xleftarrow{\$} \Pi.\mathsf{Encap}^*(\mathsf{mpk}_{r_{i,j}}, \boldsymbol{x}^*), \quad k'_{r_{i,j}} = \Pi.\mathsf{Decap}(\mathsf{sk}_{f_i}^{(r_{i,j})}, \mathsf{ct}'_{r_{i,j}}),\\ &\mathsf{ct}'_{n+r_{i,j}} = \mu_{b,r_{i,j}} + \mathsf{Ext}(k'_{r_{i,j}}, s_{r_{i,j}}).\end{aligned}$$

On the other hand, for each $r_{i,j} \in \Gamma_0$, the challenger computes

$$\begin{aligned}&(\mathsf{ct}_{r_{i,j}}, k_{r_{i,j}}) \xleftarrow{\$} \Pi.\mathsf{Encap}(\mathsf{mpk}_{r_{i,j}}, \boldsymbol{x}^*), \quad k'_{r_{i,j}} = \Pi.\mathsf{Decap}(\mathsf{sk}_{f_i}^{(r_{i,j})}, \mathsf{ct}_{r_{i,j}}),\\ &\mathsf{ct}_{n+r_{i,j}} = \mu_{b,r_{i,j}} + \mathsf{Ext}(k'_{r_{i,j}}, s_{r_{i,j}}).\end{aligned}$$

The only difference between H_2 and H_3 is the computation and usage of $\mathsf{ct}_{r_{i,j}}$ and $\mathsf{ct}'_{r_{i,j}}$ for each $r_{i,j} \in [n]\backslash\Gamma_0 = ([n]\backslash\bar{\Gamma}) \cup (\bar{\Gamma}\backslash\Gamma_0)$.

Notice that, according to the ciphertext indistinguishability of the underlying AB-wHPS Π, $\{\mathsf{ct}_{r_{i,j}}\}_{r_{i,j}\in\bar{\Gamma}\backslash\Gamma_0}$ and $\{\mathsf{ct}'_{r_{i,j}}\}_{r_{i,j}\in\bar{\Gamma}\backslash\Gamma_0}$ are computationally indistinguishable even for the adversary holding the challenge secret keys $\{\mathsf{sk}_{f_i}\}_{i\in[\omega]} := \{\mathsf{sk}_{f_i}^{(r_{i,j})}\}_{i\in[\omega],j\in[t+1]}$ such that $f_i(\boldsymbol{x}^*) = 1$. This is because in this case, each invalid ciphertext $\mathsf{ct}'_{r_{i,j}}$ can be decapsulated by only one secret key in $\{\mathsf{sk}_{f_i}^{(r_{i,j})}\}_{i\in[\omega],j\in[t+1]}$. Furthermore, $\{\mathsf{ct}_{r_{i,j}}\}_{r_{i,j}\in[n]\backslash\bar{\Gamma}}$ and $\{\mathsf{ct}'_{r_{i,j}}\}_{r_{i,j}\in[n]\backslash\bar{\Gamma}}$ are trivially computational indistinguishability, since the adversary even does not possess any secret key that could decapsulate these ciphertexts. Hence, through combining two parts together, H_2 and H_3 are indistinguishable for the adversary having the leakage information $\{h_i(\mathsf{sk}_{f_i})\}_{i\in[\omega]}$.

Notice that, in the real scenarios of ABE, the system always issues many secret keys satisfying the specific attributes, which will be used in the following decryption computation. Therefore, it is more general for us to consider polynomially bounded ω policy function f_i such that $f_i(\boldsymbol{x}^*) = 1$ in the leakage query stage.

Hybrid H_4: This hybrid is almost same to H_3, except that the challenge ciphertext is computed in the following way:

Then for each $r_{i,j} \in \bar{\Gamma} \setminus \Gamma_0$, the challenger computes

$$\mathsf{ct}'_{r_{i,j}} \xleftarrow{\$} \Pi.\mathsf{Encap}^*(\mathsf{mpk}_{r_{i,j}}, \boldsymbol{x}^*), \quad \tilde{r}_{r_{i,j}} \xleftarrow{\$} \mathcal{M},$$
$$\mathsf{ct}'_{n+r_{i,j}} = \mu_{b,r_{i,j}} + \tilde{r}_{r_{i,j}}.$$

Essentially, $\mathsf{mpk}_{r_{i,j}}, \mathsf{ct}'_{r_{i,j}}, k'_{r_{i,j}} = \Pi.\mathsf{Decap}(\mathsf{sk}_{f_i}^{(r_{i,j})}, \mathsf{ct}'_{r_{i,j}})$ and block leakage $h_i(\mathsf{sk}_{f_i}^{(r_{i,1})}, \ldots, \mathsf{sk}_{f_i}^{(r_{i,t+1})})$ are correlated variables. According to the universality of underlying AB-wHPS, we know that $k'_{r_{i,j}}$ is uniform over $\mathcal{K}$ even given $\mathsf{mpk}_{r_{i,j}}$ and $\mathsf{ct}'_{r_{i,j}}$, i.e.,

$$H_\infty(k'_{r_{i,j}}|\mathsf{mpk}_{r_{i,j}}, \mathsf{ct}'_{r_{i,j}}) = \log(|\mathcal{K}|).$$

Furthermore, since the bit-length of leakage information $h_i(\mathsf{sk}_{f_i}) = h_i(\mathsf{sk}_{f_i}^{(r_{i,1})}, \ldots, \mathsf{sk}_{f_i}^{(r_{i,t+1})})$ is ℓ, we have

$$H_\infty(k'_{r_{i,j}}|\mathsf{mpk}_{r_{i,j}}, \mathsf{ct}'_{r_{i,j}}, h_i(\mathsf{sk}_{f_i})) \geq \log(|\mathcal{K}|) - \ell.$$

Then, for a random $s_{r_{i,j}} \xleftarrow{\$} \mathcal{S}$, $\mathsf{Ext}(k'_{r_{i,j}}, s_{r_{i,j}})$ is ε-close to the uniform distribution over $\mathcal{M}$ even given $\mathsf{mpk}_{r_{i,j}}, \mathsf{ct}'_{r_{i,j}}, h_i(\mathsf{sk}_{f_i})$, since Ext is assumed to be a strong $(\log(|\mathcal{K}|) - \ell, \varepsilon)$-extractor for $\varepsilon = \mathsf{negl}(\kappa)$.

On the other hand, for each $r_{i,j} \in [n]\backslash\bar{\Gamma}$, the challenge ciphertext can be computed in the same way as that of $r_{i,j} \in \bar{\Gamma} \setminus \Gamma_0$. The outputs of the corresponding extractor indeed satisfy the statistical closeness property, following from the universality of the underlying AB-wHPSΠ. This is because in this case, the adversary even does not possess any information on the related secret keys.

As a result, combining the above two parts of arguments, H_3 and H_4 are statistically close.

Our parameter setting ensures that the number of indexes in subset Γ_0 is at most t with an overwhelming probability. Therefore, the view of the adversary (for the challenge ciphertext) in H_4 consists of at most t shares of the challenge message and $n - t$ random values. Due to the perfect hiding property of the secret sharing scheme, the adversary's view is completely independent of μ_b and b. As a result, the advantage of $\mathcal{A}$ in H_4 is 0. Finally, combining all the

above hybrids together, we conclude that the advantage of $\mathcal{A}$ in Hybrid 0 is also negligible in κ. Thus the ABE scheme $\Pi_{\mathcal{F}}$ is ℓ-leakage-resilient for $\mathcal{F}$. □

Combining Theorems 3.12 and 7.3, we obtain the following results. Assume there exists an sel-ada/sel-sel (or ada-ada/ada-sel) secure ABE scheme with the message space $\mathbb{Z}_{\bar{m}}$ for the function class $\mathcal{F} \wedge_{\|} \mathcal{G}$, where $\mathcal{G}$ is the class as in Definition 3.9 with parameters $\bar{m}, \bar{n}$, and the key-length (of the extra part, excluding the function description of f) of this underlying ABE scheme for policy function f is $s(f)$. Then the allowed leakage length of the above leakage resilient ABE scheme $\Pi_{\mathcal{F}}$ with parameters $n, \hat{t}, \omega$ as in the above paragraph setting for the function class $\mathcal{F}$ is $\ell = (\bar{n} \log \bar{m} - 2\kappa)$ and the key-length of $\Pi_{\mathcal{F}}$ for f is $|\mathsf{sk}_f| = (\hat{t}+1)(\log n + \bar{n} \log \bar{m} + |f| + s(\hat{f}_{f,g_y}))$.

Furthermore, if the secret key size $s(\hat{f}_{f,g_y})$ is succinct, i.e., $s(\hat{f}_{f,g_y}) = o(\bar{n} \log \bar{m} + |f|)$, then we can set sufficiently large $n, \bar{m}, \bar{n}$ such that $(\log n + |f|) = o(\bar{n} \log \bar{m})$. Consequently, when the underlying secret keys form a block source under each leakage function, the corresponding leakage rate of this scheme $\Pi_{\mathcal{F}}$ is $\frac{\bar{n} \log \bar{m} - 2\kappa}{\log n + \bar{n} \log \bar{m} + |f| + s(\hat{f}_{f,g_y})} = \frac{1 - \frac{2\kappa}{\bar{n} \log \bar{m}}}{1 + \frac{\log n + |f| + s(\hat{f}_{f,g_y})}{\bar{n} \log \bar{m}}} \approx 1 - o(1)$, achieving the desired optimal leakage rate.

Finally, by combining Corollary 4.8 and Theorem 7.3, we obtain the following Corollary.

Corollary 7.4 *Assuming* LWE*, for any* $S = \mathsf{poly}(\kappa)$ *and* $\omega = \mathsf{poly}(\kappa)$*, there exist* (ℓ, ω)*-leakage resilient* ABE*'s in the relative leakage model, which are*

1. *adaptively secure for* t-CNF* *functions of size up to* S*;*
2. *selectively secure for general circuits of size up to* S*.*

Moreover, when the underlying secret keys form a block source under the each leakage function, the corresponding leakage rate is $1 - o(1)$*.*

Furthermore, we can also achieve similar results in the BRM. By combining Corollary 4.8, Theorems 6.3 and 7.3, we obtain the following corollary.

Corollary 7.5 *Assuming* LWE*, for any polynomial* $S = \mathsf{poly}(\kappa)$ *and* $\omega = \mathsf{poly}(\kappa)$*, there exist* (ℓ, ω)*-leakage resilient* ABE *schemes in the* BRM*, which are*

1. *adaptively secure for* t-CNF* *functions of size up to* S*;*
2. *selectively secure for general circuits of size up to* S*.*

Moreover, when the underlying secret keys form a block source under the each leakage function, the corresponding leakage rate is $1 - o(1)$*.*

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10623-024-01358-1.

Acknowledgements We would like to thank the reviewers of PKC 2022 for their insightful advices. Qiqi Lai is supported by the National Natural Science Foundation of China (Grant Nos. 62172266, 61802241), and Henan Key Laboratory of Network Cryptography Technology (Grant No. LNCT2021-A03). Feng-Hao Liu is supported by the NSF Career Award CNS-1942400. Zhedong Wang is supported by the National Science Foundation of China (Grant No. 62202305) and the Shanghai Pujiang Program (Grant No. 22PJ1407700).

References

1. Agrawal D., Archambeault B., Rao J.R., Rohatgi P.: The EM side-channel(s). In: Kaliski B.S. Jr., Koç Ç.K., Paar C. (eds.) CHES 2002, volume 2523 of LNCS, pp. 29–45. Springer, Heidelberg (2003).

2. Agrawal S., Freeman D.M., Vaikuntanathan V.: Functional encryption for inner product predicates from learning with errors. In: Lee D.H., Wang X. (eds.) ASIACRYPT 2011, volume 7073 of LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011).
3. Agrawal S., Boneh D., Boyen X.: Efficient lattice (H)IBE in the standard model. In Gilbert [23], pp. 553–572.
4. Akavia A., Goldwasser S., Vaikuntanathan V.: Simultaneous hardcore bits and cryptography against memory attacks. In: Reingold O. (ed.) TCC 2009, volume 5444 of LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009).
5. Alwen J., Peikert C.: Generating shorter bases for hard random lattices. Theory Comput. Syst. **48**(3), 535–553 (2010).
6. Alwen J., Dodis Y., Naor M., Segev G., Walfish S., Wichs D.: Public-key encryption in the bounded-retrieval model. In Gilbert [23], pp. 113–134.
7. Alwen J., Dodis Y., Wichs D.: Leakage-resilient public-key cryptography in the bounded-retrieval model. In Halevi [28], pp. 36–54.
8. Apon D., Fan X., Liu F.-H.: Vector encoding over lattices and its applications. Cryptology ePrint Archive, Report 2017/455, (2017). http://eprint.iacr.org/2017/455
9. Bellare M., Ristenpart T.: Simulation without the artificial abort: simplified proof and improved concrete security for Waters' IBE scheme. In: Joux A. (ed.) EUROCRYPT 2009, volume 5479 of LNCS, vol. 5479, pp. 407–424. Springer, Heidelberg (2009).
10. Boneh D., Gentry C., Gorbunov S., Halevi S., Nikolaenko V., Segev G., Vaikuntanathan V., Vinayagamurthy D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen P.Q., Oswald E. (eds.) EUROCRYPT 2014, volume 8441 of LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014).
11. Brakerski Z., Goldwasser S.: Circular and leakage resilient public-key encryption under subgroup indistinguishability- (or: Quadratic residuosity strikes back). In: Rabin T. (ed.) CRYPTO 2010, volume 6223 of LNCS, vol. 6223, pp. 1–20. Springer, Heidelberg (2010).
12. Brakerski Z., Kalai Y.T., Katz J., Vaikuntanathan V.: Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In FOCS 2010 [22], pp. 501–510.
13. Brakerski Z., Langlois A., Peikert C., Regev O., Stehlé D.: Classical hardness of learning with errors. In: Boneh D., Roughgarden T., Feigenbaum J. (eds.) 45th ACM STOC, pp. 575–584. ACM Press (2013).
14. Brakerski Z., Lombardi A., Segev G., Vaikuntanathan V.: Anonymous IBE, leakage resilience and circular security from new assumptions. In: Nielsen J.B., Rijmen V. (eds.) EUROCRYPT 2018, Part I, volume 10820 of LNCS, pp. 535–564. Springer, Heidelberg (2018).
15. Chen J., Gay R., Wee H.: Improved dual system ABE in prime-order groups via predicate encodings. In: Oswald E., Fischlin M. (eds.) EUROCRYPT 2015, Part II, volume 9057 of LNCS, pp. 595–624. Springer, Heidelberg (2015).
16. Cramer R., Shoup V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen L.R. (ed.) EUROCRYPT 2002, volume 2332 of LNCS, pp. 45–64. Springer, Heidelberg (2002).
17. Dodis Y., Ostrovsky R., Reyzin L., Smith A.D.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008).
18. Dodis Y., Goldwasser S., Kalai Y.T., Peikert C., Vaikuntanathan V.: Public-key encryption schemes with auxiliary inputs. In Micciancio [37], pp. 361–381.
19. Dodis Y., Haralambiev K., López-Alt A., Wichs D.: Cryptography against continuous memory attacks. In FOCS 2010 [22], pp. 511–520.
20. Dziembowski S.: On forward-secure storage (extended abstract). In: Dwork C. (ed.) CRYPTO 2006, volume 4117 of LNCS, vol. 4117, pp. 251–270. Springer, Heidelberg (2006).
21. Faust S., Mukherjee P., Nielsen J.B., Venturi D.: Continuous non-malleable codes. In Lindell [35], pp. 465–488.
22. Gong J., Chen J., Dong X., Cao Z., Tang S.: Extended nested dual system groups, revisited. In: Cheng C.-M., Chung K.-M., Persiano G., Yang B.-Y. (eds.) PKC 2016, Part I, volume 9614 of LNCS, pp. 133–163. Springer, Heidelberg (2016).
23. Gorbunov S., Vinayagamurthy D.: Riding on asymmetry: efficient ABE for branching programs. In: Iwata T., Cheon J.H. (eds.) ASIACRYPT 2015, Part I, volume 9452 of LNCS, pp. 550–574. Springer, Heidelberg (2015).
24. Gorbunov S., Vaikuntanathan V., Wee H.: Functional encryption with bounded collusions via multi-party computation. In Safavi-Naini and Canetti [44], pp. 162–179.
25. Haldermany J.A.: Lest we remember: cold boot attacks on encryption keys. Commun. ACM **52**(5), 91–98 (2008).

26. Hazay C., López-Alt A., Wee H., Wichs D.: Leakage-resilient cryptography from minimal assumptions. In: Johansson T., Nguyen P.Q. (eds.) EUROCRYPT 2013, volume 7881 of LNCS, vol. 7881, pp. 160–176. Springer, Heidelberg (2013).
27. Kiayias A., Liu F.-H., Tselekounis Y.: Practical non-malleable codes from l-more extractable hash functions. In: Weippl E.R., Katzenbeisser S., Kruegel C., Myers A.C., Halevi S. (eds.) ACM CCS 2016, pp. 1317–1328. ACM Press, New York (2016).
28. Kocher P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz N. (ed.) CRYPTO'96, volume 1109 of LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996).
29. Lai Q., Liu F.-H., Wang Z.: Leakage-resilient IBE/ ABE with optimal leakage rates from lattices. In: Hanaoka G., Shikata J., Watanabe Y. (eds.) PKC 2022, Part II, volume 13178 of LNCS, pp. 225–255. Springer, Heidelberg (2022).
30. Lewko A.B., Waters B.: New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Micciancio [37], pp. 455–479.
31. Lewko A.B., Rouselakis Y., Waters B.: Achieving leakage resilience through dual system encryption. In: Ishai Y. (ed.) TCC 2011, volume 6597 of LNCS, vol. 6597, pp. 70–88. Springer, Heidelberg (2011).
32. Liu F.-H., Lysyanskaya A.: Tamper and leakage resilience in the split-state model. In Safavi-Naini and Canetti [44], pp. 517–532.
33. Micciancio D., Peikert C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval D., Johansson T. (eds.) EUROCRYPT 2012, volume 7237 of LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012).
34. Naor M., Segev G.: Public-key cryptosystems resilient to key leakage. In Halevi [28], pp. 18–35.
35. Nisan N., Zuckerman D.: Randomness is Linear in Space. Academic Press, Inc., Cambridge (1996).
36. Nishimaki R., Yamakawa T.: Leakage-resilient identity-based encryption in bounded retrieval model with nearly optimal leakage-ratio. In: Lin D., Sako K. (eds.) PKC 2019, Part I, volume 11442 of LNCS, pp. 466–495. Springer, Heidelberg (2019).
37. Peikert C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher M. (ed.) 41st ACM STOC, pp. 333–342. ACM Press (2009).
38. Regev O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow H.N., Fagin R. (eds.) 37th ACM STOC, pp. 84–93. ACM Press (2005).
39. Sahai A., Waters B.R.: Fuzzy identity-based encryption. In: Cramer R. (ed), EUROCRYPT 2005, volume 3494 of LNCS, pp. 457–473. Springer, Heidelberg (2005).
40. Tsabary R.: Fully secure attribute-based encryption for t-CNF from LWE. In Boldyreva, A., Micciancio, D. (eds), CRYPTO 2019, Part I, volume 11692 of LNCS, pp. 62–85. Springer, Heidelberg (2019).
41. Vadhan S.P.: On constructing locally computable extractors and cryptosystems in the bounded storage model. In Boneh D. (ed), CRYPTO 2003, volume 2729 of LNCS, pp. 61–77. Springer, Heidelberg (2003).
42. Vadhan S.P.: Pseudorandomness. Found. Trends Theor. Comput. Sci. **7**(13), 1–336 (2012).
43. Waters B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Halevi [28], pp. 619–636.
44. Wee H.: Dual system encryption via predicate encodings. In Lindell [35], pp. 616–637.
45. Zhang L., Zhang J., Mu Y.: Novel leakage-resilient attribute-based encryption from hash proof system. Comput. J. **60**(4), 541–554, 09 (2016).
46. Zhang M., Zhang Y., Su Y., Huang Q., Mu Y.: Attribute-based hash proof system under learning-with-errors assumption in obfuscator-free and leakage-resilient environments. IEEE Syst. J. **11**(2), 1018–1026 (2017).