Predicate Encryption for Circuits from LWE

Sergey Gorbunov[⋆1], Vinod Vaikuntanathan[⋆⋆1], and Hoeteck Wee[⋆⋆⋆2]

[1] MIT
[2] ENS

Abstract. In predicate encryption, a ciphertext is associated with descriptive attribute values x in addition to a plaintext μ, and a secret key is associated with a predicate f. Decryption returns plaintext μ if and only if $f(x) = 1$. Moreover, security of predicate encryption guarantees that an adversary learns nothing about the attribute x or the plaintext μ from a ciphertext, given arbitrary many secret keys that are not authorized to decrypt the ciphertext individually.

We construct a leveled predicate encryption scheme for all circuits, assuming the hardness of the subexponential learning with errors (LWE) problem. That is, for any polynomial function $d = d(\lambda)$, we construct a predicate encryption scheme for the class of all circuits with depth bounded by $d(\lambda)$, where λ is the security parameter.

1 Introduction

Predicate encryption [BW07,SBC+07,KSW08] is a new paradigm for public-key encryption that supports searching on encrypted data. In predicate encryption, ciphertexts are associated with descriptive attribute values x in addition to plaintexts μ, secret keys are associated with a predicate f, and a secret key decrypts the ciphertext to recover μ if and only if $f(x) = 1$. The security requirement for predicate encryption enforces privacy of x and the plaintext even amidst multiple secret key queries: an adversary holding secret keys for different query predicates learns nothing about the attribute x and the plaintext (apart from the fact that x does not satisfy any of the query predicates) if none of them is individually authorized to decrypt the ciphertext.

[⋆] Email: `sergeyg@mit.edu`. Supported in part by the Northrop Grumman Cybersecurity Research Consortium (CRC) and by a Microsoft PhD Fellowship.

[⋆⋆] Email: `vinodv@csail.mit.edu`. Research supported in part by DARPA Grant number FA8750- 11-2-0225, NSF Awards CNS-1350619 and CNS-1413920, an Alfred P. Sloan Research Fellowship, the Northrop Grumman Cybersecurity Research Consortium (CRC), Microsoft Faculty Fellowship, and a Steven and Renee Finn Career Development Chair from MIT.

[⋆⋆⋆] E-mail: `wee@di.ens.fr`. CNRS, INRIA and Columbia University. Supported in part by NSF Award CNS-1445424, ERC project aSCEND (639554), the Alexander von Humboldt Foundation and a Google Faculty Research Award.

Motivating applications. We begin with several motivating applications for predicate encryption [BW07,SBC$^+$07]:

– For inspecting recorded log files for network intrusions, we would encrypt network flows labeled with a set of attributes from the network header, such as the source and destination addresses, port numbers, time-stamp, and protocol numbers. We could then issue auditors with restricted secret keys that can only decrypt the network flows that fall within a particular range of IP addresses and some specific time period.
– For credit card fraud investigation, we would encrypt credit card transactions labeled with a set of attributes such as time, costs and zipcodes. We could then issue investigators with restricted secret keys that decrypt transactions over $1,000 which took place in the last month and originated from a particular range of zipcodes.
– For anti-terrorism investigation, we would encrypt travel records labeled with a set of attributes such as travel destination and basic traveller data. We could then issue investigators with restricted secret keys that match certain suspicious travel patterns.
– For online dating, we would encrypt personal profiles labeled with dating preferences pertaining to age, height, weight, salary and hobbies. Secret keys are associated with specific attributes and can only decrypt profiles for which the attributes match the dating preferences.

In all of these examples, it is important that unauthorized parties do not learn the contents of the ciphertexts, nor of the meta-data associated with the ciphertexts, such as the network header or dating preferences. On the other hand, it is often okay to leak the meta-data to authorized parties. We stress that privacy of the meta-data is an additional security requirement provided by predicate encryption but not by the related and weaker notion of attribute-based encryption (ABE) [SW05,GPSW06]; the latter only guarantees the privacy of the plaintext μ and not the attribute x.

Utility and expressiveness. The utility of predicate encryption is intimately related to the class of predicates for which we could create secret keys. Ideally, we would like to support the class of all circuits. Over the past decade, substantial advances were made for the weaker primitive of ABE, culminating most recently in schemes supporting any policy computable by general circuits [GVW13,BGG$^+$14] under the standard LWE assumption [Reg09]. However, the state-of-the-art for predicate encryption is largely limited to very simple functionalities related to computing an inner product [BW07,SBC$^+$07,KSW08,AFV11,GMW15].

1.1 Our Contributions

In this work, we substantially advance the state of the art to obtain predicate encryption for all circuits (c.f. Figure 1):

Theorem (informal). Under the LWE assumption, there exists a predicate encryption scheme for all circuits, with succint ciphertexts and secret keys independent of the size of the circuit.

As with prior LWE-based ABE for circuits [GVW13,BGG$^+$14], to support circuits of depth d, the parameters of the scheme grow with $\text{poly}(d)$, and we require sub-exponential $n^{\Omega(d)}$ hardness of the LWE assumption. In addition, the security guarantee is selective, but can be extended to adaptive security via complexity leveraging [BB04].

Privacy guarantees. The privacy notion we achieve is a *simulation-based* variant of "attribute-hiding" from the literature [SBC$^+$07,OT10,AFV11]. That is, we guarantee privacy of the attribute x and the plaintext μ against collusions holding secret keys for functions f such that $f(x) = 0$. An even stronger requirement would be to require privacy of x even against authorized keys corresponding to functions f where $f(x) = 1$; in the literature, this stronger notion is referred to as "full attribute-hiding" [BW07,KSW08]. This stronger requirement is equivalent to "full-fledged" functional encryption [BSW11], for which we cannot hope to achieve simulation-based security for all circuits as achieved in this work [BSW11,AGVW13].

Relation to prior works. Our result subsumes all prior works on predicate encryption under standard cryptographic assumptions, apart from a few exceptions pertaining to the inner product predicate [BW07,KSW08,OT12]. These results achieve a stronger security notion for predicate encryption, known as full (or strong) security (please refer to Section 3.1, and the full version for definitions).

In a recent break-through work, Garg et al. [GGH$^+$13b] gave a beautiful candidate construction of functional encryption (more general primitive than predicate encryption) for arbitrary circuits. However, the construction relies on "multi-linear maps" [GGH13a,CLT13,GGH15], for which we have few candidates and which rely on complex intractability assumptions that are presently poorly understood and not extensively studied in the literature. It remains an intriguing open problem to construct a functional encryption scheme from a standard assumption, such as LWE.

In contrast, if we consider functional encryption with *a-priori bounded collusions size* (that is, the number of secret keys any collusion of adversaries may obtain is fixed by the scheme at the setup phase), then it is possible to obtain functional encryption for general circuits under a large class of standard assumptions [SS10,GVW12,GKP$^+$13]. This notion is *weaker* than standard notion of functional encryption, yet remains very meaningful for many applications.

1.2 Overview of Our Construction

Our starting point is the work of Goldwasser, Kalai, Popa, Vaikuntanathan and Zeldovich [GKP$^+$13] who show how to convert an attribute-based encryption

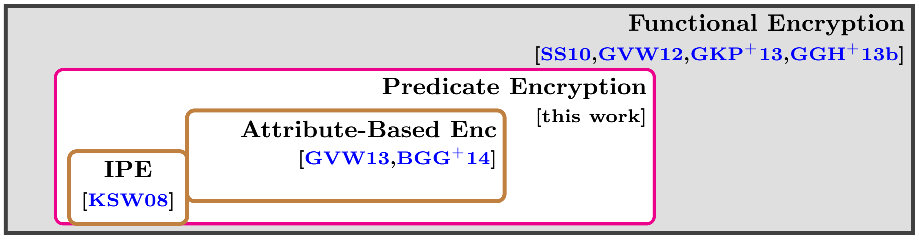

Fig. 1. State of the art in functional encryption. The white region refers to functionalities for which we have constructions under standard cryptographic assumptions like LWE or decisional problems in bilinear groups: these functionalities include inner product encryption (IPE), attribute-based encryption for general circuits (ABE) and predicate encryption for general circuits. The grey region refers to functionalities beyond predicate encryption for which we only have constructions for weaker security notions like bounded collusions, or under non-standard cryptographic assumptions like obfuscation or multilinear maps.

	Interface	Security Guarantee given sk_f
ABE	$\mathsf{Enc}(x,\mu)$	μ is secret iff $f(x)=0$
		x is always public
PE	$\mathsf{Enc}(x,\mu)$	(x,μ) is secret iff $f(x)=0$
FE	$\mathsf{Enc}(x)$	user learns only $f(x)$

Fig. 2. Comparison of the security guarantees provided by attribute-based (ABE), predicate (PE) and functional encryption (FE), where secret keys are associated with a Boolean function f; the main distinction lies in how much information about x is potentially leaked to the adversary. The main distinction between ABE and PE is that x is always public in ABE, but remains secret in PE when the user is not authorized to decrypt. The main distinction between PE and FE is that x always remains hidden (even when $f(x)=1$) and hence the user only learns the output of the computation of f on x.

(ABE) scheme into a *single key secure* functional encryption (FE) scheme. Recall that in an attribute-based encryption scheme [GPSW06], a ciphertext is associated with a descriptive value (a public "attribute") x and plaintext μ, and it hides μ, but not x. The observation of Goldwasser et al. [GKP+13] is to hide x by encrypting it using a fully homomorphic encryption (FHE) scheme [Gen09,BV11b], and then using the resulting FHE ciphertext as the public "attribute" in an ABE scheme for general circuits [GVW13,BGG+14]. This has the dual benefit of guaranteeing privacy of x, while at the same time allowing homomorphic computation of predicates f on the encryption of x.

This initial idea quickly runs into trouble. The decryptor who is given the predicate secret key for f and a predicate encryption of (x,μ) can indeed compute an *FHE encryption* of $f(x)$. However, the decryption process is confronted with a decision, namely whether to release the message μ or not, and this decision depends on whether the *plaintext* $f(x)$ is 0 or 1.[3] Clearly, resolving this conundrum requires obtaining $f(x)$, which requires knowledge of the FHE secret key. Goldwasser et al. [GKP+13] solved this by employing a (single use) Yao garbling of the FHE decryption circuit, however this limited them to obtaining *single key secure* predicate/functional encryption schemes.[4]

Our first key idea is to embed the FHE secret key as part of the attributes in the ABE ciphertext. That is, in order to encrypt a plaintext μ with attributes x in the predicate encryption scheme, we first choose a symmetric key $\mathsf{fhe.sk}$ for the FHE scheme, encrypt x into a FHE ciphertext $\hat{x}$, and encrypt μ using the ABE scheme with $(\mathsf{fhe.sk}, \hat{x})$ as the attributes to obtain an ABE ciphertext ct. Our predicate encryption ciphertext is then given by

$$(\hat{x}, \mathsf{ct})$$

To generate the predicate secret key for a function f, one simply generates the ABE secret key for the function g that takes as input $(\mathsf{fhe.sk}, \hat{x})$ and computes

$$g(\mathsf{fhe.sk}, \hat{x}) = \mathsf{FHE.Dec}(\mathsf{fhe.sk}; \mathsf{FHE.Eval}(f, \hat{x}))$$

That is, g first homomorphically computes a FHE encryption of $f(x)$, and then decrypts it using the FHE secret key to output $f(x)$.

At first glance, this idea evokes strong and conflicting emotions as it raises two problems. The first pertains to correctness: we can no longer decrypt the ciphertext since the ABE decryption algorithm needs to know all of the attributes ($\hat{x}$ and $\mathsf{fhe.sk}$), but $\mathsf{fhe.sk}$ is missing. The second pertains to security: the ABE ciphertext ct is not guaranteed to protect the privacy of the attributes, and could leak all of $\mathsf{fhe.sk}$ which together with $\hat{x}$ would leak all of x. Solving both of these problems seems to require designing a predicate encryption scheme from scratch!

Our next key observation is that the bulk of the computation in g, namely the homomorphic evaluation of the function f, is performed on the *public* attribute $\hat{x}$. The only computation performed on the secret value $\mathsf{fhe.sk}$ is FHE decryption which is a fairly lightweight computation. In particular, with all known FHE schemes [Gen09,BV11b,BV11a,BGV12,GSW13,BV14,AP14],

[3] In fact, there is a syntactic mismatch since $\hat{f}(\cdot)$ is not a predicate, as it outputs an FHE ciphertext.

[4] A reader familiar with [GKP+13] might wonder whether replacing single-use garbled circuits in their construction with reusable garbled circuits (also from [GKP+13]) might remove this limitation. We remark that this does not seem possible, essentially because the construction in [GKP+13] relies crucially on the simplicity of computing garbled inputs from the "garbling key". In particular, in Yao's garbled circuit scheme, the garbling key is (many) pairs of "strings" L_0 and L_1, and a garbling of an input bit b is simply L_b. This fits perfectly with the semantics of ABE (rather, a variant termed two-input ABE in [GKP+13]) that releases one of two possible "messages" L_0 or L_1 depending on the outcome of a computation. In contrast, computing a garbled input in the reusable garbling scheme is a more complex and randomized function of the garbling key, and does not seem to align well with the semantics of ABE.

decryption corresponds to computing an inner product followed by a threshold function. Furthermore, we do know how to construct lattice-based predicate encryption schemes for threshold of inner product [AFV11,GMW15]. We stress that the latter do not correspond to FHE decryption since the inner product is computed over a vector in the ciphertext and one in the key, whereas FHE decryption requires computing an inner product over two vectors in the ciphertext; nonetheless, we will build upon the proof techniques in achieving attribute-hiding in [AFV11,GMW15] in the proof of security.

In other words, if we could enhance ABE with a modicum of secrecy so that it can perform a heavyweight computation on public attributes followed by a lightweight privacy-preserving computation on *secret* attributes, we are back in business. Our first contribution is to define such an object, that we call *partially hiding predicate encryption.*

Partially Hiding Predicate Encryption. We introduce the notion of partially hiding predicate encryption (PHPE), an object that interpolates between attribute-based encryption and predicate encryption (analogously to partial garbling in [IW14]). In PHPE, the ciphertext, encrypting message μ, is associated with an attribute (x, y) where x is private but y is always public. The secret key is associated with a function f, and decryption succeeds iff $f(x, y) = 1$. On the one extreme, considering a dummy x or functions f that ignore x and compute on y, we recover attribute-based encryption. On the other end, considering a dummy y or functions f that ignore y and compute on x, we recover predicate encryption.

We will be interested in realizing PHPE for functions ϕ of the form $\phi(x, y) = g(x, h(y))$ for some functions g and h where h may perform arbitrary heavy-weight computation on the public y and g only performs light-weight computation on the private x. Mapping back to our discussion, we would like to achieve PHPE for the "evaluate-then-decrypt" class of functions, namely where g is the FHE decryption function, h is the FHE evaluation function, x is the FHE secret key, and y is the FHE ciphertext. In general, we would like g to be simple and will allow h to be complex. It turns out that we can formalize the observation above, namely that PHPE for this class of functions gives us a predicate encryption scheme. The question now becomes: can we construct PHPE schemes for the "evaluate-then-decrypt" class of functions?

Assuming the subexponential hardness of learning with errors (LWE), we show how to construct a partially hiding predicate encryption for the class of functions $f : \mathbb{Z}_q^t \times \{0,1\}^\ell \to \{0,1\}$ of the form

$$f_\gamma(\mathbf{x}, \mathbf{y}) = \mathsf{IP}_\gamma(\mathbf{x}, h(\mathbf{y})),$$

where $h : \{0,1\}^\ell \to \{0,1\}^t$, $\gamma \in \mathbb{Z}_q$, and $\mathsf{IP}_\gamma(\mathbf{x}, \mathbf{z}) = 1$ iff $\langle \mathbf{x}, \mathbf{z} \rangle = \left(\sum_{i \in [t]} \mathbf{x}[i] \cdot \mathbf{z}[i] \right) = \gamma \mod q$.

This is almost what we want, but not quite. Recall that FHE decryption in many recent schemes [BV11b,BGV12,GSW13,BV14,AP14] is a function that checks whether an inner product of two vectors in $\mathbb{Z}_q^t$ (one of which could be over $\{0,1\}^t$) lies in a certain range. Indeed, if $\mathbf{z} \in \{0,1\}^t$ is an encryption of 1 and $\mathbf{x} \in \mathbb{Z}_q^t$ is the secret key, we know that $\langle \mathbf{x}, \mathbf{z} \rangle \in [q/2 - B, q/2 + B] \pmod q$, where B is the noise range. Applying the so-called "modulus reduction" [BV11b] transformation to all these schemes, we can assume that this range is polynomial in size.

In other words, we will manage to construct a partially hiding PE scheme for the function

$$f_\gamma(\mathbf{x}, \mathbf{y}) : \langle \mathbf{x}, h(\mathbf{y}) \rangle \overset{?}{=} \gamma \pmod q$$

whereas we need a partially hiding PE scheme for the FHE decryption function which is

$$f'_R(\mathbf{x}, \mathbf{y}) : \langle \mathbf{x}, h(\mathbf{y}) \rangle \overset{?}{\in} R \pmod q$$

where R is the polynomial size range $[q/2 - B, q/2 + B]$ from above. How do we reconcile this disparity?

The "Lazy OR" Trick. The solution, called the "lazy OR trick" [SBC$^+$07,GMW15] is to publish secret keys for all functions f_γ for $\gamma \in R := [q/2 - B, q/2 + B]$. This will indeed allow us to test if the FHE decryption of the evaluated ciphertext is 1 (and reveal the message μ if it is), but it is also worrying. Publishing these predicate secret keys for the predicates f_γ reveals more information than whether $\langle \mathbf{x}, h(\mathbf{y}) \rangle \overset{?}{\in} R$. In particular, it reveals what $\langle \mathbf{x}, h(\mathbf{y}) \rangle$ is. This means that an authorized key would leak partial information about the attribute, which we do allow for predicate encryption. On the other hand, for an unauthorized key where the FHE decryption is 0, each of these $f_\gamma, \gamma \in R$ is also an unauthorized key in the PHPE and therefore leaks no information about the attribute. This extends to the collection of keys in R since the PHPE is secure against collusions. For simplicity, we assume in the rest of this overview that FHE decryption corresponds to exactly to inner product.

Asymmetry to the Rescue: Constructing Partially Hiding PE. Our final contribution is the construction of a partially hiding PE for the function class $f_\gamma(\mathbf{x}, \mathbf{y})$ above. We will crucially exploit the fact that f_γ computes an inner product on the private attribute $\mathbf{x}$. There are two challenges here: first, we need to design a decryption algorithm that knows f_γ and $\mathbf{y}$ but not $\mathbf{x}$ (this is different from decryption in ABE where the algorithm also knows $\mathbf{x}$); second, show that the ciphertext does not leak too much information about $\mathbf{x}$. We use the fully key-homomorphic encryption techniques developed by Boneh et al [BGG$^+$14] in the context of constructing an "arithmetic" ABE scheme. The crucial observation about the ABE scheme of [BGG$^+$14] is that while it was not designed to hide the attributes, it can be made to partially hide them in exactly the way we want. In particular, the scheme allows us to carry out an inner product of a public attribute vector (corresponding to the evaluated FHE ciphertext) and a private attribute vector (corresponding to the FHE secret key $\mathsf{fhe.sk}$), thanks to an inherent asymmetry in homomorphic evaluation of a multiplication gate on ABE ciphertexts. More concretely, in the homomorphic evaluation of a ciphertext for a multiplication gate in [BGG+14], the decryption algorithm works even if one of the attribute remains private, and for addition gates, the decryption algorithms works even if both attributes remain private. This addresses the first challenge of a decryption algorithm that is oblivious to $\mathbf{x}$. For the second challenge of security, we rely on techniques from inner product predicate encryption [AFV11] to prove the privacy of $\mathbf{x}$ Note that in the latter, the inner product is computed over a vector in the ciphertext and one in the key, whereas in our scheme, the inner product is computed over two vectors in the ciphertext. Interestingly, the proof still goes through since the ciphertext in the ABE [BGG+14] has the same structure as the ciphertext in [AFV11]. We refer the reader to Section 3.2 for a detailed overview of the partial hiding PE, and to Section 4 for an overview of how we combine the partial hiding PE with FHE to obtain our main result.

Finally, we remark that exploiting asymmetry in multiplication has been used in fairly different contexts in both FHE [GSW13,BV14] and in ABE [GVW13,GV14]. In [GSW13] and in this work, the use of asymmetry was crucial for realizing the underlying cryptographic primitive; whereas in [GVW13,BV14,GV14], asymmetry was used to reduce the noise growth during homomorphic evaluation, thereby leading to quantitative improvements in the underlying assumptions and hence improved efficiency.

1.3 Discussion

Comparison with other approaches. The two main alternative approaches for realizing predicate and functional encryption both rely on multi-linear maps either implicitly, or explicitly. The first is to use indistinguishability obfuscation as in [GGH+13b], and the second is to extend the dual system encryption framework to multi-linear maps [Wat09,GGHZ14]. A crucial theoretical limitation of these approaches is that they all rely on non-standard assumptions; we have few candidates for multi-linear maps [GGH13a,CLT13,GGH15] and the corresponding assumptions are presently poorly understood and not extensively studied in cryptanalysis, and in some cases, broken [CHL+15]. In particular, the latest attack in [CHL+15] highlight the importance of obtaining constructions and developing techniques that work under standard cryptographic assumptions, as is the focus of this work.

Barriers to functional encryption from LWE. We note the two main barriers to achieving full-fledged functional encryption from LWE using our framework. First, the lazy conjunction approach to handle threshold inner product for FHE decryption leaks the exact inner product and therefore cannot be used to achieve full attribute-hiding. Second, we do not currently know of a fully attribute-hiding inner product encryption scheme under the LWE assumption, although we do know how to obtain such schemes under standard assumptions in bilinear groups [OT12,KSW08].

2 Preliminaries

We refer the reader to the full version for the background on lattices.

2.1 Fully-Homomorphic Encryption

We present a fairly minimal definition of fully homomorphic encryption (FHE) which is sufficient for our constructions. A leveled homomorphic encryption scheme is a tuple of polynomial-time algorithms $(\mathsf{HE.KeyGen}, \mathsf{HE.Enc}, \mathsf{HE.Eval}, \mathsf{HE.Dec})$:

– **Key generation.** $\mathsf{HE.KeyGen}(1^\lambda, 1^d, 1^k)$ is a probablistic algorithm that takes as input the security parameter λ, a depth bound d and message length k and outputs a secret key sk.
– **Encryption.** $\mathsf{HE.Enc}(\mathsf{sk}, \mu)$ is a probabilistic algorithm that takes as input sk and a message $\mu \in \{0,1\}^k$ and outputs a ciphertext ct.
– **Homomorphic evaluation.** $\mathsf{HE.Eval}(f, \mathsf{ct})$ is a deterministic algorithm that takes as input a boolean circuit $C : \{0,1\}^k \to \{0,1\}$ of depth at most d and a ciphertext ct and outputs another ciphertext ct'.
– **Decryption.** $\mathsf{HE.Dec}(\mathsf{sk}, \mathsf{ct}')$ is a deterministic algorithm that takes as input sk and ciphertext ct' and outputs a bit.

Correctness. We require perfect decryption correctness with respect to homomorphically evaluated ciphertexts: namely for all λ, d, k and all $\mathsf{sk} \leftarrow \mathsf{HE.KeyGen}(1^\lambda, 1^d, 1^k)$, all $\mu \in \{0,1\}^k$ and for all boolean circuits $C : \{0,1\}^k \to \{0,1\}$ of depth at most d:

$$\Pr\Big[\mathsf{HE.Dec}(\mathsf{sk},\ \mathsf{HE.Eval}(C,\ \mathsf{HE.Enc}(\mathsf{sk}, \mu))) = C(\mu)\Big] = 1$$

where the probablity is taken over $\mathsf{HE.Enc}$ and $\mathsf{HE.KeyGen}$.

Security. We require semantic security for a single ciphertext: namely for every stateful p.p.t. adversary $\mathcal{A}$ and for all $d, k = \mathrm{poly}(\lambda)$, the following quantity

$$\Pr\left[b = b' : \begin{array}{l} \mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda, 1^d, 1^k); \\ (\mu_0, \mu_1) \leftarrow \mathcal{A}(1^\lambda, 1^d, 1^k); \\ b \xleftarrow{\$} \{0,1\}; \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{sk}, \mu_b); \\ b' \leftarrow \mathcal{A}(\mathsf{ct}) \end{array} \right] - \frac{1}{2}$$

is negligible in λ.

FHE from LWE We will rely on an instantiation of FHE from the LWE assumption:

Theorem 2.1 (FHE from LWE [BV11b,BGV12,GSW13,BV14,AP14]). *There is a FHE scheme* $\mathsf{HE.KeyGen}, \mathsf{HE.Enc}, \mathsf{HE.Eval}, \mathsf{HE.Dec}$ *that works for any* q *with* $q \geq O(\lambda^2)$ *with the following properties:*

– $\mathsf{HE.KeyGen}$ *outputs a secret key* $\mathsf{sk} \in \mathbb{Z}_q^t$ *where* $t = \mathrm{poly}(\lambda)$*;*
– $\mathsf{HE.Enc}$ *outputs a ciphertext* $\mathsf{ct} \in \{0,1\}^\ell$ *where* $\ell = \mathrm{poly}(k, d, \lambda, \log q)$*;*
– $\mathsf{HE.Eval}$ *outputs a ciphertext* $\mathsf{ct}' \in \{0,1\}^t$*;*
– *for any boolean circuit of depth* d*,* $\mathsf{HE.Eval}(C, \cdot)$ *is computed by a boolean circuit of depth* $\mathrm{poly}(d, \lambda, \log q)$*.*
– $\mathsf{HE.Dec}$ *on input* $\mathsf{sk}, \mathsf{ct}'$ *outputs a bit* $b \in \{0,1\}$*. If* ct' *is an encryption of* 1 *then*

$$\sum_{i=1}^{t} \mathsf{sk}[i] \cdot \mathsf{ct}'[i] \in [\lfloor q/2 \rfloor - B, \lfloor q/2 \rfloor + B]$$

for some fixed $B = \mathrm{poly}(\lambda)$*. Otherwise, if* ct' *is an encryption of* 0*, then*

$$\sum_{i=1}^{t} \mathsf{sk}[i] \cdot \mathsf{ct}'[i] \notin [\lfloor q/2 \rfloor - B, \lfloor q/2 \rfloor + B];$$

– *security relies on* $\mathsf{dLWE}_{\Theta(t),q,\chi}$*.*

We highlight several properties of the above scheme: (1) the ciphertext is a bit-string, (2) the bound B is a polynomial independent of q (here, we crucially exploit the new results in [BV14] together with the use of leveled bootstrapping)[5], (3) the size of normal ciphertexts is independent of the size of the circuit (this is the typical compactness requirement).

3 Partially Hiding Predicate Encryption

3.1 Definitions

We introduce the notation of partially hiding predicate encryption (PHPE), which interpolates attribute-based encryption and predicate encryption (analogously to partial garbling in [IW14]). In PHPE, the ciphertext, encrypting message μ, is associated with an attribute (x, y) where x is private but y is always public. The secret key is associated with a predicate C, and decryption succeeds iff $C(x, y) = 1$. The requirement is that a collusion learns nothing about (x, μ) if none of them is individually authorized to decrypt the ciphertext. Attribute-based encryption corresponds to the setting where x is empty, and predicate encryption corresponds to the setting where y is empty. We refer the reader to the full version for the standard notion of predicate encryption.

Looking ahead to our construction, we show how to:

– construct PHPE for a restricted class of circuits that is "low complexity" with respect to x and allows arbitrarily polynomial-time computation on y;
– bootstrap this PHPE using FHE to obtain PE for all circuits.

[5] Recall that no circular security assumption needs to be made for leveled bootstrapping.

Syntax. A Partially-Hiding Predicate Encryption scheme $\mathcal{PHPE}$ for a pair of input-universes $\mathcal{X}, \mathcal{Y}$, a predicate universe $\mathcal{C}$, a message space $\mathcal{M}$, consists of four algorithms $(\mathsf{PH.Setup}, \mathsf{PH.Enc}, \mathsf{PH.Keygen}, \mathsf{PH.Dec})$:

$\mathsf{PH.Setup}(1^\lambda, \mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M}) \rightarrow (\mathsf{ph.mpk}, \mathsf{ph.msk})$. The setup algorithm gets as input the security parameter λ and a description of $(\mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M})$ and outputs the public parameter $\mathsf{ph.mpk}$, and the master key $\mathsf{ph.msk}$.

$\mathsf{PH.Enc}(\mathsf{ph.mpk}, (x, y), \mu) \rightarrow \mathsf{ct}_y$. The encryption algorithm gets as input $\mathsf{ph.mpk}$, an attribute $(x, y) \in \mathcal{X} \times \mathcal{Y}$ and a message $\mu \in \mathcal{M}$. It outputs a ciphertext ct_y.

$\mathsf{PH.Keygen}(\mathsf{ph.msk}, C) \rightarrow \mathsf{sk}_C$. The key generation algorithm gets as input $\mathsf{ph.msk}$ and a predicate $C \in \mathcal{C}$. It outputs a secret key sk_C.

$\mathsf{PH.Dec}((\mathsf{sk}_C, C), (\mathsf{ct}_y, y)) \rightarrow \mu$. The decryption algorithm gets as input the secret key sk_C, a predicate C, and a ciphertext ct_y and the public part of the attribute y. It outputs a message $\mu \in \mathcal{M}$ or $\perp$.

Correctness. We require that for all $\mathsf{PH.Setup}(1^\lambda, \mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M}) \rightarrow (\mathsf{ph.mpk}, \mathsf{ph.msk})$, for all $(x, y, C) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{C}$, for all $\mu \in \mathcal{M}$,

– if $C(x, y) = 1$, $\Pr\left[\mathsf{PH.Dec}((\mathsf{sk}_C, C), (\mathsf{ct}_y, y)) = \mu\right] \geq 1 - \mathrm{negl}(\lambda)$,
– if $C(x, y) = 0$, $\Pr\left[\mathsf{PH.Dec}((\mathsf{sk}_C, C), (\mathsf{ct}_y, y)) = \perp\right] \geq 1 - \mathrm{negl}(\lambda)$,

where the probabilities are taken over $\mathsf{sk}_C \leftarrow \mathsf{PH.Keygen}(\mathsf{ph.msk}, C)$, $\mathsf{ct}_y \leftarrow \mathsf{PH.Enc}(\mathsf{ph.mpk}, (x, y), \mu)$ and coins of $\mathsf{PH.Setup}$.

Definition 3.1 (PHPE Attribute-Hiding). *Fix* $(\mathsf{PH.Setup}, \mathsf{PH.Enc}, \mathsf{PH.Keygen}, \mathsf{PH.Dec})$. *For every stateful p.p.t. adversary* Adv, *and a p.p.t. simulator* Sim, *consider the following two experiments:*

```
$\underline{\exp^{\mathsf{real}}_{\mathcal{PHPE},\mathrm{Adv}}(1^\lambda)}$:

1: $(x, y) \leftarrow \mathrm{Adv}(1^\lambda, \mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M})$
2: $(\mathsf{ph.mpk}, \mathsf{ph.msk}) \leftarrow$
   $\mathsf{PH.Setup}(1^\lambda, \mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M})$
3: $\mu \leftarrow \mathrm{Adv}^{\mathsf{PH.Keygen}(\mathsf{msk}, \cdot)}(\mathsf{ph.mpk})$
4: $\mathsf{ct}_y \leftarrow \mathsf{PH.Enc}(\mathsf{ph.mpk}, (x, y), \mu)$
5: $\alpha \leftarrow \mathrm{Adv}^{\mathsf{PH.Keygen}(\mathsf{ph.msk}, \cdot)}(\mathsf{ct}_y)$
6: Output $(x, y, \mu, \alpha)$
```

```
$\underline{\exp^{\mathsf{ideal}}_{\mathcal{PHPE},\mathrm{Sim}}(1^\lambda)}$:

1: $(x, y) \leftarrow \mathrm{Adv}(1^\lambda, \mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M})$
2: $(\mathsf{ph.mpk}, \mathsf{ph.msk}) \leftarrow$
   $\mathsf{PH.Setup}(1^\lambda, \mathcal{X}, \mathcal{Y}, \mathcal{C}, \mathcal{M})$
3: $\mu \leftarrow \mathrm{Adv}^{\mathsf{PH.Keygen}(\mathsf{ph.msk}, \cdot)}(\mathsf{ph.mpk})$
4: $\mathsf{ct}_y \leftarrow \mathrm{Sim}(\mathsf{mpk}, y, 1^{|x|}, 1^{|\mu|})$
5: $\alpha \leftarrow \mathrm{Adv}^{\mathsf{PH.Keygen}(\mathsf{msk}, \cdot)}(\mathsf{ct}_y)$
6: Output $(x, y, \mu, \alpha)$
```

We say an adversary Adv *is admissible if all oracle queries that it makes* $C \in \mathcal{C}$ *satisfy* $C(x, y) = 0$. *The Partially-Hiding Predicate Encryption scheme* $\mathcal{PHPE}$ *is then said to be* attribute-hiding *if there is a p.p.t. simulator* Sim *such that for every stateful p.p.t. adversary* Adv, *the following two distributions are computationally indistinguishable:*

$$\left\{\exp^{\mathsf{real}}_{\mathcal{PHPE},\mathrm{Adv}}(1^\lambda)\right\}_{\lambda \in \mathbb{N}} \overset{c}{\approx} \left\{\exp^{\mathsf{ideal}}_{\mathcal{PHPE},\mathrm{Sim}}(1^\lambda)\right\}_{\lambda \in \mathbb{N}}$$

Remarks. We point out some remarks of our definition (SIM-AH) when treated as a regular predicate encryption (i.e. the setting where y is empty; see the full version for completeness) and how it compares to other definitions in the literature.

– We note the simulator for the challenge ciphertext gets y but not x; this captures the fact that y is public whereas x is private. In addition, the simulator is not allowed to program the public parameters or the secret keys. In the ideal experiment, the simulator does not explicitly learn any information about x (apart from its length); nonetheless, there is implicit leakage about x from the key queries made by an admissible adversary. Finally, we note that we can efficiently check whether an adversary is admissible.
– Our security notion is "selective", in that the adversary "commits" to (x, y) before it sees $\mathsf{ph.mpk}$. It is possible to bootstrap selectively-secure scheme to full security using standard complexity leveraging arguments [BB04,GVW13], at the price of a $2^{|x|}$ loss in the security reduction.
– Our definition refers to a single challenge message, but the definition extends readily to a setting with multiple challenge messages. Moreover, our definition composes in that security for a single message implies security with multiple messages (see the full version). The following remarks refer to many messages setting.
– We distinguish between two notions of indistinguishability-based (IND) definitions used in the literature: attribute-hiding (IND-AH)[6] and strong attribute-hiding (IND-SAH)[7] [BW07,SBC+07,KSW08,AFV11]. In the IND-AH, the adversary should not be able to distinguish between two pairs of attributes/messages given that it is restricted to queries which do not decrypt the challenge ciphertext (See the full version for details). It is easy to see that our SIM-AH definition is stronger than IND-AH. Furthermore, IND-SAH also ensures that adversary cannot distinguish between the attributes even when it is allowed to ask for queries that decrypt the messages (in this case, it must output $\mu_0 = \mu_1$). Our SIM-AH definition is weaker than IND-SAH, since we explicitly restrict the adversary to queries that do not decrypt the challenge ciphertext.
– In the context of arbitrary predicates, *strong* variants of definitions (that is, IND-SAH and SIM-SAH) are equivalent to security notions for functional encryption (the simulation definition must be adjusted to give the simulated the outputs of the queries). However, the strong variant of notion (SIM-SAH) is impossible to realize for many messages [BSW11,AGVW13]. We refer the reader to the full version for a sketch of the impossibility. Hence, SIM-AH is the best-possible simulation security for predicate encryption which we realize in this work. The only problem which we leave open is to realize IND-SAH from standard LWE.

[6] Sometimes also referred as weak attribute-hiding.
[7] Sometimes also referred as full attribute-hiding.

3.2 Our Construction

We refer the reader to the full version for the complete description of our construction. Below, we provide an overview.

Overview. We construct a partially hiding predicate encryption for the class of predicate circuits $C : \mathbb{Z}_q^t \times \{0,1\}^\ell \to \{0,1\}$ of the form $\widehat{C} \circ \mathsf{IP}_\gamma$ where $\widehat{C} : \{0,1\}^\ell \to \{0,1\}^t$ is a boolean circuit of depth d, $\gamma \in \mathbb{Z}_q$, and

$$(\widehat{C} \circ \mathsf{IP}_\gamma)(\mathbf{x}, \mathbf{y}) = \mathsf{IP}_\gamma(\mathbf{x}, \widehat{C}(\mathbf{y})),$$

where $\mathsf{IP}_\gamma(\mathbf{x}, \mathbf{z}) = 1$ iff $\langle \mathbf{x}, \mathbf{z} \rangle = \Big(\sum_{i \in [t]} \mathbf{x}[i] \cdot \mathbf{z}[i] \Big) = \gamma \mod q$. We refer to circuit IP as the generic inner-product circuit of two vectors.

Looking ahead, $\widehat{C}$ corresponds to FHE evaluation of an arbitrary circuit C, whereas IP_γ corresponds to roughly to FHE decryption; in the language of the introduction in Section 1, $\widehat{C}$ corresponds to heavy-weight computation h, whereas IP_γ corresponds to light-weight computation g.

The scheme. The public parameters are matrices

$$\big(\mathbf{A},\ \mathbf{A}_1, \ldots, \mathbf{A}_\ell,\ \mathbf{B}_1, \ldots, \mathbf{B}_t \big)$$

An encryption corresponding to the attribute $(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}_q^t \times \{0,1\}^\ell$ is a GPV ciphertext (an LWE sample) corresponding to the matrix

$$\big[\mathbf{A} \mid \mathbf{A}_1 + \mathbf{y}[1] \cdot \mathbf{G} \mid \cdots \mid \mathbf{A}_\ell + \mathbf{y}[\ell] \cdot \mathbf{G} \mid \mathbf{B}_1 + \mathbf{x}[1] \cdot \mathbf{G} \mid \cdots \mid \mathbf{B}_t + \mathbf{x}[t] \cdot \mathbf{G} \big]$$

To decrypt the ciphertext given $\mathbf{y}$ and a key for $\widehat{C} \circ \mathsf{IP}_\gamma$, we apply the BGGHNSVV algorithm to first transform the first part of the ciphertext into a GPV ciphertext corresponding to the matrix

$$\big[\mathbf{A} \mid \mathbf{A}_{\widehat{C}_1} + \mathbf{z}[1] \cdot \mathbf{G} \mid \cdots \mid \mathbf{A}_{\widehat{C}_t} + \mathbf{z}[t] \cdot \mathbf{G} \big]$$

where $\widehat{C}_i$ is the circuit computing the i'th bit of $\widehat{C}$ and $\mathbf{z} = \widehat{C}(\mathbf{y}) \in \{0,1\}^t$. Next, observe that

$$-\Big((\mathbf{A}_{\widehat{C}_i} + \mathbf{z}[i] \cdot \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{B}_i) \Big) + \mathbf{z}[i] \cdot \Big(\mathbf{B}_i + \mathbf{x}[i] \cdot \mathbf{G} \Big) = -\mathbf{A}_{\widehat{C}_i} \mathbf{G}^{-1}(\mathbf{B}_i) + \mathbf{x}[i] \cdot \mathbf{z}[i] \cdot \mathbf{G}.$$

Summing over i, we have

$$\sum_{i=1}^{\ell} -\Big((\mathbf{A}_{\widehat{C}_i} + \mathbf{z}[i] \cdot \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{B}_i) \Big) + \mathbf{z}[i] \cdot \Big(\mathbf{B}_i + \mathbf{x}[i] \cdot \mathbf{G} \Big) = \mathbf{A}_{\widehat{C} \circ \mathsf{IP}} + \langle \mathbf{x}, \mathbf{z} \rangle \cdot \mathbf{G}$$

where

$$\mathbf{A}_{\widehat{C} \circ \mathsf{IP}} := -\Big(\mathbf{A}_{\widehat{C}_1} \mathbf{G}^{-1}(\mathbf{B}_1) + \cdots + \mathbf{A}_{\widehat{C}_t} \mathbf{G}^{-1}(\mathbf{B}_t) \Big).$$

Therefore, given only the public matrices and $\mathbf{y}$ (but not $\mathbf{x}$), we may transform the ciphertext into a GPV ciphertext corresponding to the matrix

$$\left[\,\mathbf{A} \mid \mathbf{A}_{\widehat{C}\circ\mathsf{IP}} + \langle \mathbf{x}, \mathbf{z}\rangle \cdot \mathbf{G}\,\right].$$

The secret key corresponding to $\widehat{C} \circ \mathsf{IP}_\gamma$ is essentially a "short basis" for the matrix

$$\left[\,\mathbf{A} \mid \mathbf{A}_{\widehat{C}\circ\mathsf{IP}} + \gamma \cdot \mathbf{G}\,\right]$$

which can be sampled using a short trapdoor $\mathbf{T}$ of the matrix $\mathbf{A}$.

Proof strategy. There are two main components to the proof. Fix the selective challenge attribute $\mathbf{x}, \mathbf{y}$. First, we will simulate the secret keys without knowing the trapdoor for the matrix $\mathbf{A}$: here, we rely on the simulated key generation for the ABE [BGG+14]. Roughly speaking, we will need to generate a short basis for the matrix

$$\left[\,\mathbf{A} \mid \mathbf{A}\mathbf{R}_{\widehat{C}\circ\mathsf{IP}} + (\gamma - \widehat{C} \circ \mathsf{IP}(\mathbf{x}, \mathbf{y})) \cdot \mathbf{G}\,\right]$$

where $\mathbf{R}_{\widehat{C}\circ\mathsf{IP}}$ is a small-norm matrix known to the simulator. Now, whenever $\widehat{C} \circ \mathsf{IP}(\mathbf{x}, \mathbf{y}) \neq \gamma$ as is the case for admissible adversaries, we will be able to simultae secret keys using the puncturing techniques in [ABB10,AFV11,MP12]

Next, we will show that the attribute $\mathbf{x}$ is hidden in the challenge ciphertext. Here, we adopt the proof strategy for attribute-hiding inner product encryption in [AFV11,GMW15]. In the proof, we simulate the matrices $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_t$ using

$$\mathbf{A}, \mathbf{A}\mathbf{R}'_1 - \mathbf{x}[1]\mathbf{G}, \ldots, \mathbf{A}\mathbf{R}'_t - \mathbf{x}[t]\mathbf{G}$$

where $\mathbf{R}'_1, \ldots, \mathbf{R}'_t \overset{\$}{\leftarrow} \{\pm 1\}^{m\times m}$. In addition, we simulate the corresponding terms in the challenge ciphertext by $\mathbf{c}, \mathbf{c}^\intercal\mathbf{R}'_1, \ldots, \mathbf{c}^\intercal\mathbf{R}'_t$, where $\mathbf{c}$ is a uniformly random vector, which we switched from $\mathbf{A}^\intercal\mathbf{s}+\mathbf{e}$ using the LWE assumption. Here we crucially rely on the fact that switched to simulation of secret keys without knowing the trapdoor of $\mathbf{A}$. Going further, once $\mathbf{c}$ is random, we can switch back to simulating secret keys using the trapdoor $\mathbf{T}$. Hence, the secret keys now do not leak any information about $\mathbf{R}'_1, \ldots, \mathbf{R}'_t$. Therefore, we may then invoke the left-over hash lemma to argue that $\mathbf{x}$ is information-theoretically hidden.

4 Predicate Encryption for Circuits

In this section, we present our main construction of predicate encryption for circuits by bootstrapping on top of the partially-hiding predicate encryption. That is,

- We construct a Predicate Encryption scheme $\mathcal{PE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ for boolean predicate family $\mathcal{C}$ bounded by depth d over k bit inputs.

starting from

– an FHE scheme $\mathcal{FHE} = (\mathsf{HE.KeyGen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$ with properties as described in Section 2.1. Define ℓ as the size of the initial ciphertext encrypting k bit messages, and t as the size of the FHE secret key and evaluated ciphertext vectors;
– a partially-hiding predicate encryption scheme $\mathcal{PHPE} = (\mathsf{PH.Setup}, \mathsf{PH.Keygen}, \mathsf{PH.Enc}, \mathsf{PH.Dec})$ for the class $\mathcal{C}_{\mathsf{PHPE}}$ of predicates bounded by some depth parameter $d' = \mathrm{poly}(d, \lambda, \log q)$. Recall that

$$(\widehat{C} \circ \mathsf{IP}_\gamma)(\mathbf{x} \in \mathbb{Z}_q^t, \mathbf{y} \in \{0,1\}^t) = 1 \text{ iff } \left(\sum_{i \in [t]} \mathbf{x}[i] \cdot \widehat{C}(\mathbf{y})[i]\right) = \gamma \mod q$$

where $\widehat{C} : \{0,1\}^\ell \to \{0,1\}^t$ is a circuit of depth at most d'.

Overview. At a high level, the construction proceeds as follows:

– the $\mathcal{PE}$ ciphertext corresponding to an attribute $\mathbf{a} \in \{0,1\}^k$ is a $\mathcal{PHPE}$ ciphertext corresponding to an attribute $(\mathsf{fhe.sk}, \mathsf{fhe.ct})$ where $\mathsf{fhe.sk} \xleftarrow{\$} \mathbb{Z}_q^t$ is private and $\mathsf{fhe.ct} := \mathsf{HE.Enc}(\mathbf{a}) \in \{0,1\}^\ell$ is public;
– the $\mathcal{PE}$ secret key for a predicate $C : \{0,1\}^k \to \{0,1\} \in \mathcal{C}$ is a collection of $2B+1$ $\mathcal{PHPE}$ secret keys for the predicates $\{\widehat{C} \circ \mathsf{IP}_\gamma : \mathbb{Z}_q^t \times \{0,1\}^\ell \to \{0,1\}\}_{\gamma = \lfloor q/2 \rfloor - B, \ldots, \lfloor q/2 \rfloor + B}$ where $\widehat{C} : \{0,1\}^\ell \to \{0,1\}$ is the circuit:

$$\widehat{C}(\mathsf{fhe.ct}) := \mathsf{HE.Eval}(\mathsf{fhe.ct}, C),$$

so $\widehat{C}$ is a circuit of depth at most $d' = \mathrm{poly}(d, \lambda, \log q)$;
– decryption works by trying all possible $2B+1$ secret keys.

Note that the construction relies crucially on the fact that B (the bound on the noise in the FHE evaluated ciphertexts) is polynomial. For correctness, observe that for all $C, \mathbf{a}$:

$$\begin{aligned}
&C(\mathbf{a}) = 1 \\
&\Leftrightarrow \mathsf{HE.Dec}(\mathsf{fhe.sk}, \mathsf{HE.Eval}(C, \mathsf{fhe.ct})) = 1 \\
&\Leftrightarrow \exists\, \gamma \in [\lfloor q/2 \rfloor - B, \lfloor q/2 \rfloor + B] \text{ such that } \left(\sum_{i \in [t]} \mathsf{fhe.sk}[i] \cdot \mathsf{fhe.ct}[i]\right) = \gamma \mod q \\
&\Leftrightarrow \exists\, \gamma \in [\lfloor q/2 \rfloor - B, \lfloor q/2 \rfloor + B] \text{ such that } (\widehat{C} \circ \mathsf{IP}_\gamma)(\mathsf{fhe.sk}, \mathsf{fhe.ct}) = 1
\end{aligned}$$

where $\mathsf{fhe.sk}, \mathsf{fhe.ct}, \hat{C}$ are derived from $C, \mathbf{a}$ as in our construction.

4.1 Our Predicate Encryption scheme

Our construction proceeds as follows:

– $\mathsf{Setup}(1^\lambda, 1^k, 1^d)$: The setup algorithm takes the security parameter λ, the attribute length k and the predicate depth bound d.

1. Run the partially-hiding PE scheme for family $\mathcal{C}_{\mathsf{PHPE}}$ to obtain a pair of master public and secret keys:
$$(\mathsf{ph.mpk}, \mathsf{ph.msk}) \leftarrow \mathsf{PH.Setup}(1^\lambda, 1^t, 1^\ell, 1^{d'})$$
where for k-bit messages and depth d circuits: t is the length of FHE secret key, ℓ is the bit-length of the initial FHE ciphertext and d' is the bound on FHE evaluation circuit (as described at the beginning of this section).
2. Output $(\mathsf{mpk} := \mathsf{ph.mpk}, \mathsf{msk} := \mathsf{ph.msk})$.

– $\mathsf{Keygen}(\mathsf{msk}, C)$: The key-generation algorithms takes as input the master secret key msk and a predicate C. It outputs a secret key sk_C computed as follows.
1. Let $\widehat{C}(\cdot) := \mathsf{HE.Eval}(\cdot, C)$ and let $(\widehat{C} \circ \mathsf{IP}_\gamma)$ be the predicates for $\gamma = \lfloor q/2 \rfloor - B, \dots, \lfloor q/2 \rfloor + B$.
2. For all $\gamma = \lfloor q/2 \rfloor - B, \dots, \lfloor q/2 \rfloor + B$, compute
$$\mathsf{sk}_{\widehat{C} \circ \mathsf{IP}_\gamma} \leftarrow \mathsf{PH.Keygen}\big(\mathsf{ph.msk}, \widehat{C} \circ \mathsf{IP}_\gamma\big)$$
3. Output the secret key as $\mathsf{sk}_C := \big(\{\mathsf{sk}_{\widehat{C} \circ \mathsf{IP}}\}_{\gamma = \lfloor q/2 \rfloor - B, \dots, \lfloor q/2 \rfloor + B}\big)$.

– $\mathsf{Enc}(\mathsf{mpk}, \mathbf{a}, \mu)$: The encryption algorithm takes as input the public key mpk, the input attribute vector $\mathbf{a} \in \{0,1\}^k$ and message $\mu \in \{0,1\}$. It proceeds as follow.
1. Samples a fresh FHE secret key $\mathsf{fhe.sk} \in \mathbb{Z}_q^t$ by running $\mathsf{HE.KeyGen}(1^\lambda, 1^{d'}, 1^k)$.
2. Encrypt the input to obtain
$$\mathsf{fhe.ct} \leftarrow \mathsf{HE.Enc}(\mathsf{fhe.sk}, \mathbf{a}) \in \{0,1\}^\ell$$
3. Compute
$$\mathsf{ct}_{\mathsf{fhe.ct}} \leftarrow \mathsf{PH.Enc}\big(\mathsf{mpk}, (\mathsf{fhe.sk}, \mathsf{fhe.ct}), \mu\big)$$
Note that the $\mathsf{fhe.sk}$ corresponds to the hidden attribute and $\mathsf{fhe.ct}$ corresponds to the public attribute.
4. Output the ciphertext $\mathsf{ct} = (\mathsf{ct}_{\mathsf{fhe.ct}}, \mathsf{fhe.ct})$.

– $\mathsf{Dec}((\mathsf{sk}_C, C), \mathsf{ct})$: The decryption algorithm takes as input the secret key sk_C with corresponding predicate C and the ciphertext ct. If there exists $\gamma = \lfloor q/2 \rfloor - B, \dots, \lfloor q/2 \rfloor + B$ such that
$$\mathsf{PH.Dec}((\mathsf{sk}_{\widehat{C} \circ \mathsf{IP}_\gamma}, \widehat{C} \circ \mathsf{IP}_\gamma), (\mathsf{ct}_{\mathsf{fhe.ct}}, \mathsf{fhe.ct})) = \mu \neq \perp$$
then output μ. Otherwise, output $\perp$.

4.2 Correctness

Lemma 4.1. *Let $\mathcal{C}$ be a family of predicates bounded by depth d and let $\mathcal{PHPE}$ be the partially-hiding PE and $\mathcal{FHE}$ be a fully-homomorphic encryption as per scheme description. Then, our predicate encryption scheme $\mathcal{PE}$ is correct. Moreover, the size of each secret key is* $\mathrm{poly}(d, \lambda)$ *and the size of each ciphertext is* $\mathrm{poly}(d, \lambda, k)$.

We refer the reader to the full version for the proof.

4.3 Security

Theorem 4.2. *Let $\mathcal{C}$ be a family of predicates bounded by depth d and let $\mathcal{PHPE}$ be the secure partially-hiding PE and $\mathcal{FHE}$ be the secure fully-homomorphic encryption as per scheme description. Then, our predicate encryption scheme $\mathcal{PE}$ is secure.*

Proof. We define p.p.t. simulator algorithms $\mathsf{Enc}_{\mathrm{Sim}}$ and argue that its output is indistinguishable from the output of the real experiment. Let $\mathsf{PH.Enc}_{\mathrm{Sim}}$ be the p.p.t. simulator for partially-hiding predicate encryption scheme.

– $\mathsf{Enc}_{\mathrm{Sim}}(\mathsf{mpk}, 1^{|\mathbf{a}|}, 1^{|\mu|})$: To compute the encryption, the simulator does the following. It samples FHE secret key $\mathsf{fhe.sk}$ by running $\mathsf{HE.KeyGen}(1^\lambda, 1^{d'}, 1^k)$. It encrypts a zero-string $\mathsf{fhe.ct} \leftarrow \mathsf{HE.Enc}(\mathsf{fhe.sk}, \mathbf{0})$. It obtains the ciphertext as $\mathsf{ct}_{\mathsf{fhe.ct}} \leftarrow \mathsf{PH.Enc}_{\mathrm{Sim}}(\mathsf{mpk}, \mathsf{fhe.ct}, 1^{|\mathsf{fhe.sk}|}, 1^{|\mu|})$.

We now argue via a series of hybrids that the output of the ideal experiment.

– **Hybrid 0**: The real experiment.
– **Hybrid 1**: The real encryption algorithm is replaced with Enc^*, where Enc^* is an auxiliary algorithm defined below. On the high level, Enc^* computes the FHE ciphertext honestly by sampling a secret key and using the knowledge of $\mathbf{a}$. It then invokes $\mathsf{PH.Enc}_{\mathrm{Sim}}$ on the honestly generated ciphertext.
– **Hybrid 2**: The simulated experiment.

Auxiliary Algorithms. We define the auxiliary algorithm Enc^* used in Hybrid 1.

– $\mathsf{Enc}^*(\mathbf{a}, 1^{|\mu|})$: The auxiliary encryption algorithm takes as input the attribute vector $\mathbf{a}$ and message length.
  1. Sample a fresh FHE secret key $\mathsf{fhe.sk}$ by running $\mathsf{HE.KeyGen}(1^\lambda, 1^{d'}, 1^k)$.
  2. Encrypt the input attribute vector to obtain a ciphertext
$$\mathsf{fhe.ct} \leftarrow \mathsf{HE.Enc}(\mathsf{fhe.sk}, \mathbf{a}) \in \{0,1\}^\ell$$
  3. Run $\mathsf{PH.Enc}_{\mathrm{Sim}}$ on input $(\mathsf{mpk}, \mathsf{fhe.ct}, 1^{|\mathsf{fhe.sk}|}, 1^{|\mu|})$ to obtain the ciphertext $\mathsf{ct}_{\mathsf{fhe.ct}}$.

Lemma 4.3. *The output of Hybrid 0 is computationally indistinguishable from the Hybrid 1, assuming security of Partially-Hiding Predicate Encryption.*

Proof. Assume there is an adversary Adv and a distinguisher $\mathcal{D}$ that distinguishes the output $(\mathbf{a}, \mu, \alpha)$ produced in either of the two hybrids. We construct an adversary Adv$'$ and a distinguisher $\mathcal{D}'$ that break the security of the Partially-Hiding Predicate Encryption. The adversary Adv$'$ does the following.

1. Invoke the adversary Adv to obtain an attribute vector $\mathbf{a}$.
2. Sample a fresh FHE secret key $\mathsf{fhe.sk}$ using $\mathsf{HE.KeyGen}(1^\lambda, 1^{d'}, 1^k)$. Encrypt the attribute vector
$$\mathsf{fhe.ct} \leftarrow \mathsf{HE.Enc}(\mathsf{fhe.sk}, \mathbf{a})$$
and output the pair $(\mathsf{fhe.sk}, \mathsf{fhe.ct})$ as the "selective" challenge attribute.

3. Upon receiving mpk, it forwards it to Adv.
4. For each oracle query C that Adv makes which satisfies $C(\mathbf{a}) \neq 0$, Adv$'$ uses its oracle to obtain secret keys $\mathsf{sk}_{\widehat{C} \circ \mathsf{IP}_\gamma}$ for $\gamma = \lfloor q/2 \rfloor - B, \ldots, \lfloor q/2 \rfloor + B$. It outputs $\mathsf{sk}_C = \big(\{\mathsf{sk}_{\widehat{C} \circ \mathsf{IP}_\gamma}\}_{\gamma = \lfloor q/2 \rfloor - B, \ldots, \lfloor q/2 \rfloor + B}\big)$.
5. It outputs message μ that Adv produces, obtains a ciphertext $\mathsf{ct}_{\mathsf{fhe.ct}}$ and sends $\mathsf{ct} = (\mathsf{ct}_{\mathsf{fhe.ct}}, \mathsf{fhe.ct})$ back to Adv to obtain α.

We note that given Adv that is admissible, Adv$'$ is also admissible. That is, for all queries $\widehat{C} \circ \mathsf{IP}_\gamma$ that Adv$'$ makes satisfies $(\widehat{C} \circ \mathsf{IP}_\gamma)(\mathsf{fhe.sk}, \mathsf{fhe.ct}) = 0$ since $\langle \mathsf{fhe.sk}, \widehat{C}(\mathsf{fhe.ct}) \rangle \neq \gamma$ for $\gamma = \lfloor q/2 \rfloor - B, \ldots, \lfloor q/2 \rfloor + B$ by the correctness of FHE in Section 2.1 and the fact that $C(\mathbf{a}) \neq 0$. Finally, the distinguisher $\mathcal{D}'$ on input $(\mathsf{fhe.sk}, \mathsf{fhe.ct}, \mu, \alpha)$ invokes $\mathcal{D}$ and outputs whatever it outputs. Now, in Hybrid 0 the algorithms used as $\mathsf{PH.Setup}, \mathsf{PH.Keygen}, \mathsf{PH.Enc}$ which corresponds exactly to the real security game of PHPE. However, in Hybrid 1 the algorithms correspond exactly to the simulated security game. Hence, we can distinguish between the real and simulated experiments contradicting the security of PHPE scheme.

Lemma 4.4. *The output of Hybrid 1 and Hybrid 2 are computationally indistinguishable, assuming semantic security of Fully-Homomorphic Encryption Scheme.*

Proof. The only difference in Hybrids 1 and 2 is how the FHE ciphertext is produced. In one experiment, it is computed honestly by encrypting the attribute vector $\mathbf{a}$, while in the other experiment it is always an encryption of $\mathbf{0}$. Hence, we can readily construct an FHE adversary that given $\mathbf{a}$, distinguishes encryption of $\mathbf{a}$ from encryption of $\mathbf{0}$ as follows:

1. Invoke the admissible PE adversary Adv to obtain an attribute vector $\mathbf{a}$.
2. Run the honest $\mathsf{PH.Setup}$ and forwards mpk to Adv.
3. For each oracle query C that Adv makes which satisfies $C(\mathbf{a}) \neq 0$, return $\mathsf{sk}_C = \big(\{\mathsf{sk}_{\widehat{C} \circ \mathsf{IP}_\gamma}\}_{\gamma = \lfloor q/2 \rfloor - B, \ldots, \lfloor q/2 \rfloor + B}\big)$ as computed using the honest $\mathsf{PH.Keygen}$ algorithm.
4. To simulate the ciphertext, first forward the pair $(\mathbf{a}, \mathbf{0})$ to the FHE challenger to obtain a ciphertext $\mathsf{fhe.ct}$. Then, run $\mathsf{PH.Enc}_{\mathrm{Sim}}(\mathsf{mpk}, \mathsf{fhe.ct}, 1^{|\mathsf{fhe.sk}|}, 1^{\mu})$ to obtain a ciphertext $\mathsf{ct}_{\mathsf{fhe.ct}}$ and forward it to Adv
5. Finally, it runs the PE distinguisher on input $(\mathbf{a}, \mu, \alpha)$ and outputs its guess.

The lemma then follows from semantic security of the FHE completing the security proof. We also refer the reader to the full version for the summary of parameters selection.

References

ABB10. Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, pages 553–572, 2010.

AFV11. Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *ASIACRYPT*, pages 21–40, 2011.

AGVW13. Shweta Agrawal, Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption: New perspectives and lower bounds. In *CRYPTO*, 2013. Also, Cryptology ePrint Archive, Report 2012/468.

AP14. Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In *CRYPTO (I)*, pages 297–314, 2014.

BB04. Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT*, pages 223–238, 2004.

BGG$^+$14. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, pages 533–556, 2014.

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *ITCS*, pages 309–325, 2012.

BSW11. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *TCC*, pages 253–273, 2011.

BV11a. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, pages 505–524, 2011.

BV11b. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS*, pages 97–106, 2011.

BV14. Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In *ITCS*, pages 1–12, 2014.

BW07. Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554, 2007.

CHL$^+$15. Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *EUROCRYPT*, 2015. To appear. Also, Cryptology ePrint Archive, Report 2014/906.

CLT13. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *CRYPTO (1)*, pages 476–493, 2013.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

GGH13a. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, pages 1–17, 2013.

GGH$^+$13b. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49, 2013. Also, Cryptology ePrint Archive, Report 2013/451.

GGH15. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *TCC*, 2015. Also, Cryptology ePrint Archive, Report 2014/645.

GGHZ14. Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666, 2014.

GKP$^+$13. Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *STOC*, pages 555–564, 2013.

GMW15. Romain Gay, Pierrick Méaux, and Hoeteck Wee. Predicate encryption for multi-dimensional range queries from lattices. In *Public Key Cryptography*, 2015. Also, Cryptology ePrint Archive, Report 2014/965.

GPSW06. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS*, pages 89–98, 2006.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO (1)*, pages 75–92, 2013.

GV14. Sergey Gorbunov and Dhinakaran Vinayagamurthy. Riding on asymmetry: Efficient ABE for branching programs. Cryptology ePrint Archive, Report 2014/819, 2014. http://eprint.iacr.org/.

GVW12. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO*, pages 162–179, 2012.

GVW13. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554, 2013. Also, Cryptology ePrint Archive, Report 2013/337.

IW14. Yuval Ishai and Hoeteck Wee. Partial garbling schemes and their applications. In *ICALP (1)*, pages 650–662, 2014.

KSW08. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, pages 146–162, 2008.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.

OT10. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010.

OT12. Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *EUROCRYPT*, pages 591–608, 2012. Also, Cryptology ePrint Archive, Report 2011/543.

Reg09. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

SBC$^+$07. Elaine Shi, John Bethencourt, Hubert T.-H. Chan, Dawn Xiaodong Song, and Adrian Perrig. Multi-Dimensional Range Query over Encrypted Data. In *IEEE Symposium on Security and Privacy*, pages 350–364, 2007.

SS10. Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In *ACM Conference on Computer and Communications Security*, pages 463–472, 2010.

SW05. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

Wat09. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO*, pages 619–636, 2009.