Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE

Shweta Agrawal[1], Dan Boneh[2,⋆], and Xavier Boyen[3]

[1] University of Texas, Austin
[2] Stanford University
[3] Université de Liège, Belgium

Abstract. We present a technique for delegating a short lattice basis that has the advantage of keeping the lattice dimension unchanged upon delegation. Building on this result, we construct two new hierarchical identity-based encryption (HIBE) schemes, with and without random oracles. The resulting systems are very different from earlier lattice-based HIBEs and in some cases result in shorter ciphertexts and private keys. We prove security from classic lattice hardness assumptions.

1 Introduction

Hierarchical identity based encryption (HIBE) is a public key encryption scheme where entities are arranged in a directed tree [HL02, GS02]. Each entity in the tree is provided with a secret key from its parent and can delegate this secret key to its children so that a child entity can decrypt messages intended for it, or for its children, but cannot decrypt messages intended for any other nodes in the tree. This delegation process is one-way: a child node cannot use its secret key to recover the key of its parent or its siblings. We define HIBE more precisely in the next section.

The first HIBE constructions, with and without random oracles, were based on bilinear maps [GS02, BB04, BW06, BBG05, GH09, Wat09]. More recent constructions are based on hard problems on lattices [CHKP10, ABB10] where the secret key is a "short" basis B of a certain integer lattice L. To delegate the key to a child the parent creates a new lattice L' derived from L and uses B to generate a random short basis for this lattice L'. In all previous constructions the dimension of the child lattice L' is larger than the dimension of the parent lattice L. As a result, private keys and ciphertexts become longer and longer as one descends into the hierarchy.

Our results. We first propose a new delegation mechanism that operates "in place", i.e., without increasing the dimension of the lattices involved. We then use this delegation mechanism to construct two HIBE systems where the lattices used have the same dimension for all nodes in the hierarchy. Consequently, private keys and ciphertexts are the same length for all nodes in the hierarchy. Our

⋆ Supported by NSF and the Packard Foundation.

T. Rabin (Ed.): CRYPTO 2010, LNCS 6223, pp. 98–115, 2010.

first construction, in Section 4, provides an efficient HIBE system in the random oracle model. Our second construction, in Section 5, provides selective security in the standard model, namely without random oracles. We prove security of both constructions using the classic learning with errors (LWE) problem [Reg09].

To briefly explain our delegation technique, let L be a lattice in $\mathbb{Z}^m$ and let $B = \{b_1, \ldots, b_m\}$ be a short basis of L. Let R be a public non-singular matrix in $\mathbb{Z}^{m \times m}$. Observe that the set $B' := \{R\,b_1, \ldots, R\,b_m\}$ is a basis of the lattice $L' := RL$. If all entries of the matrix R are "small" scalars then the norm of the vectors in B' is not much larger than the norm of vectors in B. Moreover, using standard tools we can "randomize" the basis without increasing the norm of the vectors by much. The end result is a random short basis of L'. This idea suggests that by associating a public "low norm" matrix R to each child, the parent node can delegate its short basis B to a child by multiplying the vectors in B by the matrix R and randomizing the resulting basis. Note that since the dimension of L' is the same as the dimension of L this delegation does not increase dimensions.

The question is whether delegation is one way: can a child L' recover a short basis of the parent L? More precisely, given a "low norm" matrix R and a random short basis of L', can one construct short vectors in $R^{-1}L'$? The key ingredient in proving security is an algorithm called SampleRwithBasis that given a lattice L (for which no short basis is given) outputs a "low norm" matrix R along with a short basis for the lattice $L' = RL$. In other words, if we are allowed to choose a low norm R then we can build a delegated lattice $L' = RL$ for which a short basis is known even though no short basis is given for L. Algorithm SampleRwithBasis shows that if it were possible to use a random short basis of L' to generate short vectors in L then it would be possible to solve SVP in any lattice L — generate an $L' = RL$ with a known short basis and use that basis to generate short vectors in L. More importantly, the algorithm enables us to publish matrices R so that during the HIBE simulation certain private keys are known to the simulator while others are not. The key technical challenge is showing that these simulated matrices R are distributed as in the real system.

Comparison to other lattice-based HIBE. Table 1 shows how the HIBE systems derived from our basis delegation mechanism compare to existing lattice-based HIBE systems. In the random oracle model the construction compares favorably to other lattice-based HIBE in terms of ciphertext and private key size. In terms of computation time, the encryptor in our system computes an ℓ-wise matrix product when encrypting to an identity at depth ℓ, which is not necessary in [CHKP10]. However, this product is not message dependent and need only be computed once per identity.

Our construction in the standard model treats identities at each level as k-bit binary strings. Table 1 shows that the construction is only competitive with existing HIBEs [CHKP10, ABB10] in applications where $k < \ell$ (such as [CHK07] where $k = 1$). When $k > \ell$ the construction is not competitive due to the k^2 term in the ciphertext length (compared to $k\ell$ in [CHKP10] and ℓ in [ABB10]). Nevertheless, this HIBE is very different from the existing HIBEs and the techniques of [ABB10] can potentially be applied to improve its performance.

Table 1. A comparison of lattice-based HIBE schemes

selective secure HIBE	ciphertext length	secret key length	pub. params. length	lattice dimension	security n/α
[CHKP10] with RO	$\tilde{O}(\ell n d^2)$	$\tilde{O}(\ell^3 n^2 d^2)$	$\tilde{O}(n^2 d^2)$	$\tilde{O}(\ell d n)$	$\tilde{O}(d^d n^{d/2})$
Sec. 4 with RO	$\tilde{O}(n d^2)$	$\tilde{O}(\ell n^2 d^2)$	$\tilde{O}(n^2 d^2)$	$\tilde{O}(dn)$	$\tilde{O}((dn)^{\frac{3}{2}d})$
[CHKP10] no RO	$\tilde{O}(k\ell n d^2)$	$\tilde{O}(k^2 \ell^3 n^2 d^2)$	$\tilde{O}(k n^2 d^3)$	$\tilde{O}(k\ell dn)$	$\tilde{O}(d^d (kn)^{d/2})$
[ABB10] no RO	$\tilde{O}(\ell n d^2)$	$\tilde{O}(\ell^3 n^2 d^2)$	$\tilde{O}(n^2 d^3)$	$\tilde{O}(\ell dn)$	$\tilde{O}(d^d n^{d/2})$
Sec. 5 no RO	$\tilde{O}(k^2 n d^2)$	$\tilde{O}(k^3 \ell n^2 d^2)$	$\tilde{O}(k^3 n^2 d^3)$	$\tilde{O}(kdn)$	$\tilde{O}((kdn)^{kd+\frac{d}{2}})$

The table compares the lengths of the ciphertext, private key, and lattice dimension. We let n be the security parameter, d be the maximum hierarchy depth (determined at setup time), and ℓ be the depth of the identity in question. When appropriate we let k be the number of bits in each component of the identity. The last column shows the SVP approximation factor that needs to be hard in the worst-case for the systems to be secure (the smaller the better). We focus on selectively secure HIBE since for all known adaptive lattice HIBE security degrades exponentially in the hierarchy depth.

Relation to bilinear map constructions. The recent lattice-based IBE and HIBE systems are closely related to their bilinear map counterparts and there may exist an abstraction that unifies these constructions. While the mechanics are quite different the high level structure is similar. The construction and proof of security in [CHKP10] resembles the tree construction of Canetti et al. [CHK07]. The construction and proof of security in [ABB10] resembles the constructions of Boneh and Boyen [BB04] and Waters [Wat05]. The constructions in this paper have some relation to the HIBE of Boneh, Boyen, and Goh [BBG05], although the relation is not as direct. Waters [Wat09] recently proposed dual-encryption as a method to build fully secure HIBE systems from bilinear maps. It is a beautiful open problem to construct a lattice analog of that result using either the basis delegation in this paper or the method from [CHKP10]. It is quite possible that two lattice-based dual-encryption HIBE systems will emerge.

2 Preliminaries

Notation. Throughout the paper we say that a function $\epsilon : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is negligible if $\epsilon(n)$ is smaller than all polynomial fractions for sufficiently large n. We say that an event happens with overwhelming probability if it happens with probability at least $1 - \epsilon(n)$ for some negligible function ϵ. We say that integer vectors $v_1, \ldots, v_n \in \mathbb{Z}^m$ are $\mathbb{Z}_q$-linearly independent for prime q if they are linearly independent when reduced modulo q.

2.1 Hierarchical IBE

Recall that an Identity-Based Encryption system (IBE) consists of four algorithms [Sha85, BF01]: Setup, $\mathsf{Extract}$, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$. The Setup algorithm generates system parameters, denoted by PP, and a master key MK. The $\mathsf{Extract}$ algorithm uses the master key to extract a private key corresponding to a given identity. The encryption algorithm encrypts messages for a given identity and the decryption algorithm decrypts ciphertexts using the private key.

In a Hierarchical IBE [HL02, GS02], identities are vectors, and there is a fifth algorithm called Derive. Algorithm Derive takes an identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_k)$ at depth k and a private key $\mathsf{SK}_{\mathsf{id}|_\ell}$ of a parent identity $\mathsf{id}_{|_\ell} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ for some $\ell < k$. It outputs the private key $\mathsf{SK}_{\mathsf{id}}$ for the identity id which is distributed the same as the output of $\mathsf{Extract}$ for id.

Selective and Adaptive ID Security. The standard IBE security model of [BF01] allows an attacker to adaptively choose the identity it wishes to attack. A weaker notion of IBE called selective security [CHK07] forces the adversary to announce ahead of time the public key it will target. We use both notions, but restrict the adversary to chosen-plaintext attacks.

Security Game. We define HIBE security using a game that captures a strong privacy property called *indistinguishable from random* which means that the challenge ciphertext is indistinguishable from a random element in the ciphertext space. This property implies both semantic security and recipient anonymity, and also implies that the ciphertext hides the public parameters (PP) used to create it. For a security parameter λ, we let $\mathcal{M}_\lambda$ denote the message space and let $\mathcal{C}_\lambda$ denote the ciphertext space. The selective security game, for a hierarchy of maximum depth d, proceeds as follows.

Init: The adversary is given the maximum depth of the hierarchy d and outputs a target identity $\mathsf{id}^* = (\mathsf{I}_1^*, \ldots, \mathsf{I}_k^*), k \leq d$.

Setup: The challenger runs $\mathsf{Setup}(1^\lambda, 1^d)$ (where $d = 1$ for IBE) and gives the adversary the resulting system parameters PP.

Phase 1: The adversary adaptively issues queries on identities $\mathsf{id}_1, \mathsf{id}_2, \ldots$ where no query is for a prefix of id^*. For each query the challenger runs algorithm $\mathsf{Extract}$ to obtain a private key d_i for the public key id_i and sends d_i to the adversary.

Challenge: Once the adversary decides that Phase 1 is over it outputs a plaintext $M \in \mathcal{M}_\lambda$ on which it wishes to be challenged. The challenger chooses a random bit $r \in \{0, 1\}$ and a random ciphertext $C \in \mathcal{C}_\lambda$. If $r = 0$ it sets the challenge ciphertext to $C^* := \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}^*, M)$. If $r = 1$ it sets the challenge ciphertext to $C^* := C$. It sends C^* as the challenge to the adversary.

Phase 2: The adversary issues additional adaptive private key queries as in phase 1 and the challenger responds as before.

Guess: Finally, the adversary outputs a guess $r' \in \{0, 1\}$ and wins if $r = r'$.

We refer to such an adversary $\mathcal{A}$ as an INDr–sID-CPA adversary and define its advantage in attacking $\mathcal{E}$ as $\mathrm{Adv}_{\mathcal{E},\mathcal{A},d}(\lambda) = \big|\Pr[r = r'] - 1/2\big|$.

Definition 1. *A depth d HIBE system $\mathcal{E}$ is selective-identity, indistinguishable from random if for all INDr–sID-CPA PPT adversaries $\mathcal{A}$ the function $Adv_{\mathcal{E},\mathcal{A},d}(\lambda)$ is negligible. We say that $\mathcal{E}$ is INDr–sID-CPA secure for depth d.*

We define the adaptive-identity counterparts to the above notions by removing the Init phase from the attack game, and allowing the adversary to wait until the Challenge phase to announce the identity id^* it wishes to attack. The adversary is allowed to make arbitrary private-key queries in Phase 1 and then choose an arbitrary target id^* as long as he did not issue a private-key query for a prefix of id^* in phase 1. The resulting security notion is defined using the modified game as in Definition 1, and is denoted INDr–ID-CPA.

2.2 Statistical Distance

Let X and Y be two random variables taking values in some finite set Ω. Define the *statistical distance*, denoted $\Delta(X;Y)$, as

$$\Delta(X;Y) := \frac{1}{2}\sum_{s\in\Omega}\big|\Pr[X = s] - \Pr[Y = s]\big|$$

We say that X is δ-uniform over Ω if $\Delta(X;U_\Omega) \le \delta$ where U_Ω is a uniform random variable over Ω. Two ensembles of random variables $X(\lambda)$ and $Y(\lambda)$ are statistically close if $d(\lambda) := \Delta(X(\lambda);Y(\lambda))$ is a negligible function of λ.

2.3 Integer Lattices

Definition 2. *Let $B = \left[\, b_1 \,\middle|\, \ldots \,\middle|\, b_m \,\right] \in \mathbb{R}^{m\times m}$ be an $m \times m$ matrix whose columns are linearly independent vectors $b_1, \ldots, b_m \in \mathbb{R}^m$. The m-dimensional full-rank lattice Λ generated by B is the set,*

$$\Lambda = \mathcal{L}(B) = \left\{\, y \in \mathbb{R}^m \quad s.t. \quad \exists s \in \mathbb{Z}^m \;,\;\; y = B\,s = \sum_{i=1}^{m} s_i\, b_i \,\right\}$$

Here, we are interested in integer lattices, i.e, when L is contained in $\mathbb{Z}^m$. We let $\det(\Lambda)$ denote the determinant of Λ.

Definition 3. *For q prime, $A \in \mathbb{Z}_q^{n\times m}$ and $u \in \mathbb{Z}_q^n$, define:*

$$\begin{aligned}
\Lambda_q(A) &:= \left\{\, e \in \mathbb{Z}^m \quad s.t. \quad \exists s \in \mathbb{Z}_q^n \text{ where } A^\top s = e \pmod q \,\right\}\\
\Lambda_q^\perp(A) &:= \left\{\, e \in \mathbb{Z}^m \quad s.t. \quad A\,e = 0 \pmod q \,\right\}\\
\Lambda_q^u(A) &:= \left\{\, e \in \mathbb{Z}^m \quad s.t. \quad A\,e = u \pmod q \,\right\}
\end{aligned}$$

Observe that if $t \in \Lambda_q^u(A)$ then $\Lambda_q^u(A) = \Lambda_q^\perp(A) + t$ and hence $\Lambda_q^u(A)$ is a shift of $\Lambda_q^\perp(A)$.

2.4 The Gram-Schmidt Norm of a Basis

Let S be a set of vectors $S = \{s_1, \ldots, s_k\}$ in $\mathbb{R}^m$. We use the following standard notation:

- $\|S\|$ denotes the L_2 length of the longest vector in S, i.e. $\max_{1 \le i \le k} \|s_i\|$.
- $\tilde{S} := \{\tilde{s}_1, \ldots, \tilde{s}_k\} \subset \mathbb{R}^m$ denotes the Gram-Schmidt orthogonalization of the vectors $s_1, \ldots, s_k$ taken in that order.

We refer to $\|\widetilde{S}\|$ as the Gram-Schmidt norm of S.

Micciancio and Goldwassser [MG02] showed that a full-rank set S in a lattice Λ can be converted into a basis T for Λ with an equally low Gram-Schmidt norm.

Lemma 1 ([MG02, Lemma 7.1]). *Let Λ be an m-dimensional lattice. There is a deterministic polynomial-time algorithm that, given an arbitrary basis of Λ and a full-rank set $S = \{s_1, \ldots, s_m\}$ in Λ, returns a basis T of Λ satisfying*

$$\|\widetilde{T}\| \le \|\widetilde{S}\| \quad \textit{and} \quad \|T\| \le \|S\|\sqrt{m}/2$$

Ajtai [Ajt99] and later Alwen and Peikert [AP09] show how to sample an essentially uniform matrix $A \in \mathbb{Z}_q^{n \times m}$ with an associated basis S_A of $\Lambda_q^{\perp}(A)$ with low Gram-Schmidt norm. The following follows from Theorem 3.2 of [AP09] taking $\delta := 1/3$. The theorem produces a matrix A statistically close to uniform in $\mathbb{Z}_q^{n \times m}$ along with a short basis. Since m is so much larger than n, the matrix A is rank n with overwhelming probability and we can state the theorem as saying that A is statistically close to a uniform rank n matrix in $\mathbb{Z}_q^{n \times m}$.

Theorem 1. *Let $q \ge 3$ be odd and $m := \lceil 6n \log q \rceil$. There is a probabilistic polynomial-time algorithm $\mathsf{TrapGen}(q, n)$ that outputs a pair $(A \in \mathbb{Z}_q^{n \times m},\ S \in \mathbb{Z}^{m \times m})$ such that A is statistically close to a uniform rank n matrix in $\mathbb{Z}_q^{n \times m}$ and S is a basis for $\Lambda_q^{\perp}(A)$ satisfying*

$$\|\widetilde{S}\| \le O(\sqrt{n \log q}) \quad \textit{and} \quad \|S\| \le O(n \log q)$$

with all but negligible probability in n.

Notation: We let $\tilde{L}_{\mathrm{TG}} := O(\sqrt{n \log q})$ denote the maximum (w.h.p) Gram-Schmidt norm of a basis produced by $\mathsf{TrapGen}(q, n)$.

2.5 Discrete Gaussians

Definition 4. *Let L be a subset of $\mathbb{Z}^m$. For any vector $c \in \mathbb{R}^m$ and any positive parameter $\sigma \in \mathbb{R}_{>0}$, define:*

$$\rho_{\sigma,c}(x) = \exp\left(-\pi \frac{\|x - c\|^2}{\sigma^2}\right) \quad \textit{and} \quad \rho_{\sigma,c}(L) = \sum_{x \in L} \rho_{\sigma,c}(x)$$

The discrete Gaussian distribution over L with center c and parameter σ is

$$\forall y \in L \quad , \quad \mathcal{D}_{L,\sigma,c}(y) = \frac{\rho_{\sigma,c}(y)}{\rho_{\sigma,c}(L)}$$

For notational convenience, $\rho_{\sigma,0}$ and $\mathcal{D}_{L,\sigma,0}$ are abbreviated as ρ_σ and $\mathcal{D}_{L,\sigma}$. When $\sigma = 1$ we write ρ to denote ρ_1. □

The distribution $\mathcal{D}_{L,\sigma,c}$ will most often be defined over the lattice $L = \Lambda_q^\perp(A)$ for a matrix $A \in \mathbb{Z}_q^{n\times m}$ or over a coset $L = t + \Lambda_q^\perp(A)$ where $t \in \mathbb{Z}^m$.

Properties. The following lemma from [Pei] captures standard properties of these distributions. The first property follows from Lemma 4.4 of [MR07]. The last two properties are algorithms from [GPV08].

Lemma 2. *Let $q \geq 2$ and let A be a matrix in $\mathbb{Z}_q^{n\times m}$ with $m > n$. Let T_A be a basis for $\Lambda_q^\perp(A)$ and $\sigma \geq \|\widetilde{T_A}\| \, \omega(\sqrt{\log m})$. Then for $c \in \mathbb{R}^m$ and $u \in \mathbb{Z}_q^n$:*

1. $\Pr\left[\, x \sim \mathcal{D}_{\Lambda_q^u(A),\sigma} \;:\; \|x\| > \sqrt{m}\,\sigma \,\right] \;\leq\; \mathrm{negl}(n)$.
2. *There is a PPT algorithm* $\mathsf{SampleGaussian}(A, T_A, \sigma, c)$ *that returns $x \in \Lambda_q^\perp(A)$ drawn from a distribution statistically close to $\mathcal{D}_{\Lambda,\sigma,c}$.*
3. *There is a PPT algorithm* $\mathsf{SamplePre}(A, T_A, u, \sigma)$ *that returns $x \in \Lambda_q^u(A)$ sampled from a distribution statistically close to $\mathcal{D}_{\Lambda_q^u(A),\sigma}$, whenever $\Lambda_q^u(A)$ is not empty.*

Randomizing a basis: Cash et al. [CHKP10] show how to randomize a lattice basis (see also [GN08, Sec. 2.1]).

$\mathsf{RandBasis}(S, \sigma)$:
On input a basis S of an m-dimensional lattice $\Lambda_q^\perp(A)$ and a gaussian parameter $\sigma \geq \|\widetilde{S}\| \cdot \omega(\sqrt{\log n})$, outputs a new basis S' of $\Lambda_q^\perp(A)$ such that

- with overwhelming probability $\|\widetilde{S'}\| \leq \sigma\sqrt{m}$, and
- up to a statistical distance, the distribution of S' does not depend on S. That is, the random variable $\mathsf{RandBasis}(S, \sigma)$ is statistically close to $\mathsf{RandBasis}(T, \sigma)$ for any other basis T of $\Lambda_q^\perp(A)$ satisfying $\|\widetilde{T}\| \leq \sigma/\omega(\sqrt{\log n})$.

We briefly recall how $\mathsf{RandBasis}$ works:

1. For $i = 1, \ldots, m$, let $v \leftarrow \mathsf{SampleGaussian}(A, S, \sigma, 0)$ and
   if v is independent of $\{v_1, \ldots, v_{i-1}\}$, set $v_i \leftarrow v$, if not, repeat.
2. Convert the set of vectors $v_1, \ldots, v_m$ to a basis S' using Lemma 1 (and using some canonical basis of $\Lambda_q^\perp(A)$).
3. Output S'.

The analysis of $\mathsf{RandBasis}$ in [CHKP10] uses [Reg09, Corollary 3.16] which shows that a linearly independent set is produced in Step (1) w.h.p. after m^2 samples from $\mathsf{SampleGaussian}(A, S, \sigma, 0)$. It is not difficult to show that only $2m$ samples are needed in expectation.

2.6 Hardness Assumption

Security of all our constructions reduces to the LWE (learning with errors) problem, a classic hard problem on lattices defined by Regev [Reg09].

Definition 5. *Consider a prime q, a positive integer n, and a distribution χ over $\mathbb{Z}_q$, all public. An $(\mathbb{Z}_q, n, \chi)$-LWE problem instance consists of access to an unspecified challenge oracle $\mathcal{O}$, being, either, a noisy pseudo-random sampler $\mathcal{O}_s$ carrying some constant random secret key $s \in \mathbb{Z}_q^n$, or, a truly random sampler $\mathcal{O}_\$$, whose behaviors are respectively as follows:*

$\mathcal{O}_s$**:** *outputs samples of the form $(u_i, v_i) = \left(u_i,\ u_i^\top s + x_i\right) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where, $s \in \mathbb{Z}_q^n$ is a uniformly distributed persistent value invariant across invocations, $x_i \in \mathbb{Z}_q$ is a fresh sample from χ, and u_i is uniform in $\mathbb{Z}_q^n$.*

$\mathcal{O}_\$$**:** *outputs truly uniform random samples from $\mathbb{Z}_q^n \times \mathbb{Z}_q$.*

The $(\mathbb{Z}_q, n, \chi)$-LWE problem allows repeated queries to the challenge oracle $\mathcal{O}$. We say that an algorithm $\mathcal{A}$ decides the $(\mathbb{Z}_q, n, \chi)$-LWE problem if

$$\mathsf{LWE\text{-}adv}[\mathcal{A}] := \left| \Pr[\mathcal{A}^{\mathcal{O}_s} = 1] - \Pr[\mathcal{A}^{\mathcal{O}_\$} = 1] \right|$$

is non-negligible for a random $s \in \mathbb{Z}_q^n$.

Regev [Reg09] shows that for certain noise distributions χ, denoted $\overline{\Psi}_\alpha$, the LWE problem is as hard as the worst-case SIVP and GapSVP under a quantum reduction (see also [Pei09]). Recall that for $x \in \mathbb{R}$ the symbol $\lfloor x \rceil$ denotes the closest integer to x.

Definition 6. *For an $\alpha \in (0, 1)$ and a prime q let $\overline{\Psi}_\alpha$ denote the distribution over $\mathbb{Z}_q$ of the random variable $\lfloor q\, X \rceil \mod q$ where X is a normal random variable with mean 0 and standard deviation $\alpha/\sqrt{2\,\pi}$.*

Theorem 2 ([Reg09]). *If there exists an efficient, possibly quantum, algorithm for deciding the $(\mathbb{Z}_q, n, \overline{\Psi}_\alpha)$-LWE problem for $q > 2\sqrt{n}/\alpha$ then there exists an efficient quantum algorithm for approximating the SIVP and GapSVP problems, to within $\tilde{O}(n/\alpha)$ factors in the ℓ_2 norm, in the worst case.*

The following lemma about the distribution $\overline{\Psi}_\alpha$ will be needed to show that decryption works correctly. The proof is implicit in [GPV08, Lemma 8.2].

Lemma 3. *Let e be some vector in $\mathbb{Z}^m$ and let $y \xleftarrow{R} \overline{\Psi}_\alpha^m$. Then the quantity $|e^\top y|$ treated as an integer in $[0, q-1]$ satisfies*

$$|e^\top y| \leq \|e\|\, q\alpha\, \omega(\sqrt{\log m}) + \|e\|\sqrt{m}/2$$

with all but negligible probability in m.

As a special case, Lemma 3 shows that if $x \xleftarrow{R} \overline{\Psi}_\alpha$ is treated as an integer in $[0, q-1]$ then $|x| < q\alpha\, \omega(\sqrt{\log m}) + 1/2$ with all but negligible probability in m.

3 Basis Delegation without Dimension Increase

Let A be a matrix in $\mathbb{Z}_q^{n\times m}$ and let T_A be a "short" basis of $\Lambda_q^\perp(A)$, both given. We wish to "delegate" the basis T_A in the following sense: we want to deterministically generate a matrix B from A and a random basis T_B for $\Lambda_q^\perp(B)$ such that from A, B and T_B it is difficult to recover any short basis for $\Lambda_q^\perp(A)$. Basis delegation mechanisms were proposed by Cash et al [CHKP10] and Agrawal et al. [ABB10] where the dimension of the matrix B was larger than the dimension of the given A. In the resulting HIBE systems ciphertext and private key sizes increase as the hierarchy deepens.

Here we consider a simple delegation mechanism that does not increase the dimension. To do so we use a public matrix R in $\mathbb{Z}^{m\times m}$ where the columns of R have "low" norm. We require that R be invertible mod q. Now, define $B := AR^{-1}$ in $\mathbb{Z}_q^{n\times m}$ and observe that B has the same dimension as A. We show how to build a "short" basis of $\Lambda_q^\perp(B)$ from which it is difficult to recover a short basis of A. In the next section we use this to build new HIBE systems.

We begin by defining distributions on matrices whose columns are low norm vectors. We then define the basis delegation mechanism.

Distributions on low norm matrices. We say that a matrix R in $\mathbb{Z}^{m\times m}$ is **$\mathbb{Z}_q$-invertible** if $R \bmod q$ is invertible as a matrix in $\mathbb{Z}_q^{m\times m}$. Our construction makes use of $\mathbb{Z}_q$-invertible matrices R in $\mathbb{Z}^{m\times m}$ where all the columns of R are "low norm".

Definition 7. *Define* $\sigma_{\text{R}} := \tilde{L}_{\text{TG}}\,\omega(\sqrt{\log m}) = \sqrt{n\log q}\cdot\omega(\sqrt{\log m})$.
We let $\mathcal{D}_{m\times m}$ *denote the distribution on matrices in* $\mathbb{Z}^{m\times m}$ *defined as*

$$\left(\mathcal{D}_{\mathbb{Z}^m,\sigma_{\text{R}}}\right)^m \quad \textit{conditioned on the resulting matrix being } \mathbb{Z}_q\textit{-invertible}$$

Algorithm SampleR(1^m). The following simple algorithm samples matrices in $\mathbb{Z}^{m\times m}$ from a distribution that is statistically close to $\mathcal{D}_{m\times m}$.

1. Let T be the canonical basis of the lattice $\mathbb{Z}^m$.
2. For $i = 1,\ldots,m$ do $r_i \xleftarrow{R} \mathsf{SampleGaussian}(\mathbb{Z}^m, T, \sigma_{\text{R}}, 0)$.
3. If R is $\mathbb{Z}_q$-invertible, output R; otherwise repeat step 2.

In the full version we show that step 2 will need to be repeated fewer than two times in expectation for prime q.

3.1 Basis Delegation: Algorithm BasisDel(A, R, T_A, σ)

We now describe a simple basis delegation algorithm that does not increase the dimension of the underlying matrices.

Inputs:

$$\begin{aligned}&\text{a rank } n \text{ matrix } A \text{ in } \mathbb{Z}_q^{n\times m},\\ &\text{a } \mathbb{Z}_q\text{-invertible matrix } R \text{ in } \mathbb{Z}^{m\times m} \text{ sampled from } \mathcal{D}_{m\times m}\\ &\qquad\qquad \text{(or a product of such)},\\ &\text{a basis } T_A \text{ of } \Lambda_q^\perp(A),\\ &\text{and a parameter } \sigma \in \mathbb{R}_{>0}.\end{aligned} \tag{1}$$

Output: Let $B := AR^{-1}$ in $\mathbb{Z}_q^{n\times m}$. The algorithm outputs a basis T_B of $\Lambda_q^\perp(B)$.
Algorithm BasisDel(A, R, T_A, σ)works as follows:

1. Let $T_A = \{a_1, \ldots, a_m\} \subseteq \mathbb{Z}^m$. Calculate $T'_B := \{Ra_1, \ldots, Ra_m\} \subseteq \mathbb{Z}^m$. Observe that T'_B is a set of independent vectors in $\Lambda_q^\perp(B)$.
2. Use Lemma 1 to convert T'_B into a basis T''_B of $\Lambda_q^\perp(B)$. The algorithm in the lemma takes as input T'_B and an arbitrary basis of $\Lambda_q^\perp(B)$ and outputs a basis T''_B whose Gram-Schmidt norm is no more than that of T'_B.
3. Call RandBasis(T''_B, σ) and output the resulting basis T_B of $\Lambda_q^\perp(B)$.

The following theorem shows that BasisDel produces a random basis of $\Lambda_q^\perp(B)$ whose Gram-Schmidt norm is bounded as a function of $\|\widetilde{T_A}\|$. The proof is given in the full version.

Theorem 3. *Using the notation in (1), suppose R is sampled from $\mathcal{D}_{m\times m}$ and σ satisfies*

$$\sigma > \|\widetilde{T_A}\| \cdot \sigma_{\mathrm{R}} \sqrt{m}\, \omega(\log^{3/2} m) \ .$$

Let T_B be the basis of $\Lambda_q^\perp(AR^{-1})$ output by BasisDel.
Then T_B is distributed statistically close to the distribution RandBasis(T, σ) where T is an arbitrary basis of $\Lambda_q^\perp(AR^{-1})$ satisfying $\|\widetilde{T}\| < \sigma/\omega(\sqrt{\log m})$. If R is a product of ℓ matrices sampled from $\mathcal{D}_{m\times m}$ then the bound on σ degrades to $\sigma > \|\widetilde{T_A}\| \cdot \left(\sigma_{\mathrm{R}} \sqrt{m}\, \omega(\log^{1/2} m)\right)^\ell \cdot \omega(\log m)$.

When R is a product for ℓ matrices sampled from $\mathcal{D}_{m\times m}$ then for the smallest possible σ in Theorem 3 we obtain that w.h.p

$$\|\widetilde{T_B}\| \ / \ \|\widetilde{T_A}\| \leq \left(m\, \omega(\log m)\right)^\ell \ \sqrt{m}\, \omega(\log m) \ .$$

This quantity is the minimum degradation in basis quality as we delegate across ℓ levels of the HIBE hierarchy.

3.2 The Main Simulation Tool: Algorithm SampleRwithBasis(A)

All our proofs of security make heavy use of an algorithm SampleRwithBasis that given a random rank n matrix A in $\mathbb{Z}_q^{n\times m}$ as input generates a "low-norm" matrix R (i.e., a matrix sampled from $\mathcal{D}_{m\times m}$) along with a short basis for $\Lambda_q^\perp(AR^{-1})$.

Algorithm SampleRwithBasis(A)**.** Let $a_1, \ldots, a_m \in \mathbb{Z}_q^n$ be the m columns of the matrix $A \in \mathbb{Z}_q^{n\times m}$.

1. Run TrapGen(q, n) to generate a random rank n matrix $B \in \mathbb{Z}_q^{n\times m}$ and a basis T_B of $\Lambda_q^\perp(B)$ such that $\|\widetilde{T_B}\| \leq \tilde{L}_{\mathrm{TG}} = \sigma_{\mathrm{R}}/\omega(\sqrt{\log m})$.
2. for $i = 1, \ldots, m$ do:
   (2a) sample $r_i \in \mathbb{Z}^m$ as the output of SamplePre$(B, T_B, a_i, \sigma_{\mathrm{R}})$, then $Br_i = a_i \bmod q$ and r_i is sampled from a distribution statistically close to $\mathcal{D}_{\Lambda_q^{a_i}(B), \sigma_{\mathrm{R}}}$.
   (2b) repeat step (2a) until r_i is $\mathbb{Z}_q$ linearly independent of $r_1, \ldots, r_{i-1}$.

3. Let $R \in \mathbb{Z}^{m \times m}$ be the matrix whose columns are $r_1, \ldots, r_m$.
   Then R has rank m over $\mathbb{Z}_q$. Output R and T_B.

By construction $BR = A \bmod q$ and therefore $B = AR^{-1} \bmod q$. Hence, the basis T_B is a short basis of $\Lambda_q^{\perp}(AR^{-1})$. It remains to show that R is sampled from a distribution close to $\mathcal{D}_{m \times m}$.

Theorem 4. *Let $m > 2n \log q$ and $q > 2$ a prime. For all but at most a q^{-n} fraction of rank n matrices A in $\mathbb{Z}_q^{n \times m}$ algorithm $\mathsf{SampleRwithBasis}(A)$ outputs a matrix R in $\mathbb{Z}^{m \times m}$ sampled from a distribution statistically close to $\mathcal{D}_{m \times m}$. The generated basis T_B of $\Lambda_q^{\perp}(AR^{-1})$ satisfies $\|\widetilde{T_B}\| \leq \sigma_{\mathrm{R}}/\omega(\sqrt{\log m})$ with overwhelming probability.*

The bound on $\|\widetilde{T_B}\|$ is from Theorem 1. The difficult part of the proof is arguing that R is sampled from a distribution statistically close to $\mathcal{D}_{m \times m}$. The proof is based on a detailed analysis of the distribution from which R is chosen and is given in the full version of the paper.

4 An HIBE in the Random-Oracle Model

Our first construction is a depth d HIBE secure in the random oracle model. In the next section we describe an HIBE selectively secure in the standard model.

To encrypt a message m for identity id, the encryptor builds a matrix F_{id} and encrypts m using the dual Regev public key system (described in [GPV08, sec. 7]) using F_{id} as the public key. The matrix F_{id} is built by multiplying a fixed matrix A, specified in the public parameters, by ℓ "low norm" square matrices generated by a random oracle H described in (2) below.

At level ℓ, let $\mathsf{id} = (\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_\ell) \in (\{0,1\}^*)^\ell$, where $\ell \in [d]$. We assume the availability of a hash function H that outputs matrices in $\mathbb{Z}^{m \times m}$:

$$H \;:\; (\{0,1\}^*)^{\leq d} \;\rightarrow\; \mathbb{Z}_q^{m \times m} \;:\; \mathsf{id} \;\mapsto\; H(\mathsf{id}) \sim \mathcal{D}_{m \times m} \tag{2}$$

where the requirement is that, over the choice of the random oracle H, the output $H(\mathsf{id})$ is distributed as $\mathcal{D}_{m \times m}$ (as in Definition 7). In practice, the hash function H can be built from a "standard" random function $h : (\{0,1\}^*)^{\leq d} \rightarrow \{0,1\}^t$ by using h as a coin generator for the sampling process in Algorithm $\mathsf{SampleR}(1^m)$. This method however is not indefferentiable in the sense of [CDMP05] and the analysis requires that H itself be a random oracle.

4.1 Construction

The system uses a number of parameters that will be set in Section 4.2. The parameters n, m and q are fixed across the levels of the hierarchy. In addition, we have two level-dependent parameters: a guassian parameter $\bar{\sigma} = (\sigma_1, \ldots, \sigma_d)$ and a noise parameter $\bar{\alpha} = (\alpha_1, \ldots, \alpha_d)$.

For an identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ and $1 \leq k \leq \ell$ we use $\mathsf{id}_{|k}$ to denote the vector $(\mathsf{id}_1, \ldots, \mathsf{id}_k)$. Now, for a hierarchy of maximum depth d the scheme works as follows:

Setup$(1^n, 1^d)$ On input a security parameter n and maximum depth d:

1. Invoke $\mathsf{TrapGen}(q, n)$ to generate a uniformly random matrix $A \in \mathbb{Z}_q^{n \times m}$ and a short basis $T_A = [a_1 | \dots | a_m] \in \mathbb{Z}^{m \times m}$ for $\Lambda_q^\perp(A)$.
2. Generate a uniformly random vector $u_0 \in \mathbb{Z}_q^n$.
3. Output the public parameters PP and master key MK given by,

$$\mathsf{PP} = \begin{pmatrix} A, & u_0 \end{pmatrix} \qquad \mathsf{MK} = \begin{pmatrix} T_A \end{pmatrix}$$

Derive$(\mathsf{PP}, \mathsf{SK}_{\mathsf{id}|\ell}, \mathsf{id})$**:** On input public parameters PP, a secret key $\mathsf{SK}_{\mathsf{id}|\ell}$ corresponding to a "parent" identity $\mathsf{id}_{|\ell} = (\mathsf{id}_1, \dots, \mathsf{id}_\ell)$, and a "child" identity $\mathsf{id} = (\mathsf{id}_1, \dots, \mathsf{id}_\ell, \dots \mathsf{id}_k)$ where $k \le d$ do:

1. Let $R_{\mathsf{id}|\ell} = H(\mathsf{id}_{|\ell}) \cdots H(\mathsf{id}_{|2})\, H(\mathsf{id}_{|1}) \in \mathbb{Z}^{m \times m}$ and $F_{\mathsf{id}|\ell} = A\, R_{\mathsf{id}|\ell}^{-1}$ in $\mathbb{Z}_q^{n \times m}$. Then $\mathsf{SK}_{\mathsf{id}|\ell}$ is a short basis for $\Lambda_q^\perp(F_{\mathsf{id}|\ell})$.
2. Compute $R = H(\mathsf{id}_{|k}) \cdots H(\mathsf{id}_{|\ell+1}) \in \mathbb{Z}^{m \times m}$ and set $F_{\mathsf{id}} = F_{\mathsf{id}|\ell}\, R^{-1}$.
3. Evaluate $S' \leftarrow \mathsf{BasisDel}(F_{\mathsf{id}|\ell}, R, \mathsf{SK}_{\mathsf{id}|\ell}, \sigma_k)$ to obtain a short random basis for $\Lambda_q^\perp(F_{\mathsf{id}})$.
4. Output the delegated private key $\mathsf{SK}_{\mathsf{id}} = S'$.

Algorithm $\mathsf{Extract}(\mathsf{MK}, \mathsf{id})$ works the same way by running $\mathsf{Derive}(\mathsf{PP}, \mathsf{MK}, \mathsf{id})$ where $F_{\mathsf{id}|0} = A$ and $\mathsf{SK}_{\mathsf{id}|0} = \mathsf{MK}$.

Encrypt$(\mathsf{PP}, \mathsf{id}, b)$**:** On input public parameters PP, a recipient identity id of depth $|\mathsf{id}| = \ell$, and a message bit $b \in \{0, 1\}$:

1. Compute $R_{\mathsf{id}} \leftarrow H(\mathsf{id}_{|\ell}) \dots H(\mathsf{id}_{|2})\, H(\mathsf{id}_{|1})$ in $\mathbb{Z}^{m \times m}$.
2. Compute the encryption matrix $F_{\mathsf{id}} \leftarrow A\, R_{\mathsf{id}}^{-1}$ in $\mathbb{Z}_q^{n \times m}$.
3. Now encrypt the message using Regev's dual public key encryption (as defined in [GPV08, sec. 7]) using F_{id} as the public key. To do so,
   (a) Pick a uniformly random vector $s \xleftarrow{R} \mathbb{Z}_q^n$.
   (b) Choose noise vectors $x \xleftarrow{\overline{\Psi}_{\alpha_\ell}} \mathbb{Z}_q$ and $y \xleftarrow{\overline{\Psi}_{\alpha_\ell}^m} \mathbb{Z}_q^m$. ($\overline{\Psi}_\alpha$ is as in def. 6)
   (c) Output the ciphertext,

$$\mathsf{CT} \;=\; \begin{pmatrix} \; c_0 = u_0^\top s + x + b \lfloor \frac{q}{2} \rfloor \,, & c_1 = F_{\mathsf{id}}^\top s + y \; \end{pmatrix} \;\in\; \mathbb{Z}_q \times \mathbb{Z}_q^m$$

Decrypt$(\mathsf{PP}, \mathsf{SK}_{\mathsf{id}}, \mathsf{CT})$**:** On input public parameters PP, a private key $\mathsf{SK}_{\mathsf{id}}$ for an identity id of length $|\mathsf{id}| = \ell$, and a ciphertext CT:

1. Let $\tau_\ell = \sigma_\ell \sqrt{m}\, \omega(\sqrt{\log m})$ $\quad \big(\ge \|\widetilde{\mathsf{SK}_{\mathsf{id}}}\|\, \omega(\sqrt{\log m}) \big)$.
2. Construct the matrix $F_{\mathsf{id}} \in \mathbb{Z}_q^{n \times m}$ as in step (2) of $\mathsf{Encrypt}$.
3. Set $d_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(F_{\mathsf{id}}, \mathsf{SK}_{\mathsf{id}}, u_0, \tau_\ell)$. Note that $F_{\mathsf{id}}\, d_{\mathsf{id}} = u_0$ in $\mathbb{Z}_q^n$.
4. Compute $w = c_0 - d_{\mathsf{id}}^\top c_1 \in \mathbb{Z}_q$.
5. Compare w and $\lfloor \frac{q}{2} \rfloor$ treating them as integers in $[q] \subset \mathbb{Z}$: if they are close, i.e., if $\left| w - \lfloor \frac{q}{2} \rfloor \right| < \lfloor \frac{q}{4} \rfloor$ in $\mathbb{Z}$, output 1; otherwise output 0.

4.2 Parameters and Correctness

When the cryptosystem is operated as specified, during decryption of a ciphertext encrypted to an identity at level ℓ we have,

$$w \;= c_0 - d_{\mathsf{id}}^\top c_1 = b\,\lfloor \frac{q}{2} \rfloor + \underbrace{x - d_{\mathsf{id}}^\top y}_{\text{error term}}$$

Since $\|d_{\mathsf{id}}\| \le \tau_\ell \sqrt{m} = \sigma_\ell \; m \; \omega(\sqrt{\log m}\,)$ w.h.p, we have by Lemma 3 that the norm of the error term is bounded w.h.p by

$$|x - d_{\mathsf{id}}^\top y| \le q\alpha_\ell \sigma_\ell m \; \omega(\log m) + \sigma_\ell m^{3/2} \; \omega(\sqrt{\log m}) \tag{3}$$

In addition, by properties of $\mathsf{RandBasis}(\cdot, \sigma_\ell)$ the Gram-Schmidt norm of a secret key SK_ℓ at level ℓ satisfies w.h.p. $\|\widetilde{\mathsf{SK}_\ell}\| \le \sigma_\ell \sqrt{m}$. Therefore, with $\sigma_0 = \tilde{L}_{\mathrm{TG}}$, for the system to work correctly we need that:

- $\mathsf{TrapGen}$ can operate (i.e. $m > 6n \log q$),
- the error term in (3) is less than $q/5$ w.h.p
  (i.e. $\alpha_\ell < [\sigma_\ell m \omega(\log m)]^{-1}$ and $q > \sigma_\ell \, m^{3/2} \omega(\sqrt{\log m})$),
- $\mathsf{BasisDel}$ used in Derive can operate (i.e. $\sigma_\ell > \|\widetilde{\mathsf{SK}_{\ell-1}}\| \, \sigma_{\mathrm{R}} \, \sqrt{m} \, \omega(\log^{3/2} m)$ which follows from $\sigma_\ell > \sigma_{\ell-1} \, m^{3/2} \; \omega(\log^2 m)$), and
- Regev's reduction applies (i.e. $q > 2\sqrt{n}/\alpha_\ell$ for all ℓ).

To satisfy these requirements we set the parameters $(q, m, \bar{\sigma}, \bar{\alpha})$ as follows taking n to be the security parameter (and letting $\ell = 1, \ldots, d$):

$$\begin{aligned} m = 6\, n^{1+\delta} = O(dn \log n) \quad &, \quad q = m^{\frac{3}{2}d+2} \cdot \omega(\log^{2d+1} n\,) \\ \sigma_\ell = m^{\frac{3}{2}\ell+\frac{1}{2}} \cdot \omega(\log^{2\ell} n) \quad &, \quad \alpha_\ell = [\sigma_\ell \; m \; \omega(\log n)]^{-1} \end{aligned} \tag{4}$$

and round up m to the nearest larger integer and q to the nearest larger prime. Here we assume that δ is such that $n^\delta > \lceil \log q \rceil = O(d \log n)$.

Observe that since σ_ℓ is increasing with ℓ algorithm $\mathsf{Extract}$ generates the same distribution on private keys as algorithm Derive for all identities at depth greater than one, as required from our definition of HIBE.

Overall, the ciphertext size for all identities is $\tilde{O}(d^2 n)$. Security depends on the assumption that worst-case SVP cannot be solved to within a factor $q\sqrt{n} = \tilde{O}((dn)^{1.5d})$.

4.3 Security

We state the system's security against both a selective and an adaptive adversary. Selective security implies adaptive security in the random oracle model via a simple generic transformation from [BB04]. However, proving adaptive security directly gives a slightly simpler system. Recall that selective security in the random oracle model means that the attacker must commit to the target identity before issuing any type of query.

Theorem 5. *Let $\mathcal{A}$ be a PPT adversary that attacks the scheme of Section 4.1 when H is modeled as a random oracle. Let Q_H is the number of H queries made by $\mathcal{A}$ and d be the max hierarchy depth. Then there is a PPT algorithm $\mathcal{B}$ that decides LWE such that*

1. *If $\mathcal{A}$ is a selective adversary (INDr–sID-CPA) with advantage ϵ then $\epsilon \leq \mathsf{LWE\text{-}adv}[\mathcal{B}]$.*
2. *If $\mathcal{A}$ is an adaptive adversary (INDr–ID-CPA) with advantage ϵ then $\epsilon \leq \mathsf{LWE\text{-}adv}[\mathcal{B}] \cdot (dQ_H^d) + \mathrm{negl}(n)$.*

where $\mathsf{LWE\text{-}adv}[\mathcal{B}]$ is with respect to the parameters $(\mathbb{Z}_q, n, \overline{\Psi}_\alpha)$ from Section 4.2.

Proof. We prove part (2) of the theorem. The proof of part (1) is similar and a little simpler. Recall that LWE is about recognizing an oracle $\mathcal{O}$ defined in Section 2.6. We use $\mathcal{A}$ to construct an LWE algorithm $\mathcal{B}$ with advantage about ϵ/dQ_H^d.

Instance. $\mathcal{B}$ requests from $\mathcal{O}$ and receives, for each $i = 0, \ldots, m$, a fresh pair $(u_i, v_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

As the number of oracle calls is known *a priori*, the samples can be supplied non-interactively at the beginning, e.g., here in the form of an instance with $(m+1)(n+1)$ elements of $\mathbb{Z}_q$.

Setup. $\mathcal{B}$ prepares a simulated attack environment for $\mathcal{A}$ as follows.

1. Select d uniform random integer $Q_1^*, \ldots, Q_d^* \in [Q_H]$. where Q_H is the maximum number of queries to H that $\mathcal{A}$ can make.
2. Sample d random matrices $R_1^*, \ldots, R_d^* \sim \mathcal{D}_{m \times m}$ by running $R_i^* \leftarrow \mathsf{SampleR}(1^m)$ for $i = 1, \ldots, d$.
3. Assemble the random matrix $A_0 \in \mathbb{Z}_q^{n \times m}$ from m of the given LWE samples, by letting the i-th column of A_0 be the n-vector u_i for all $i = 1, \ldots, m$.
4. Choose a random $w \in [d]$ and set $A \leftarrow A_0 R_w^* \cdots R_1^*$. The matrix A is uniform in $\mathbb{Z}_q^{n \times m}$ since all the R_i^* are invertible mod q and A_0 is uniform in $\mathbb{Z}_q^{n \times m}$.
5. Publish the public parameters $\mathsf{PP} = \big(A, u_0\big)$.

Random-oracle hash queries. $\mathcal{A}$ may query the random oracle H on any identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_i)$ of its choice, adaptively, and at any time. $\mathcal{B}$ answers the Q-th such query as follows. (We assume w.l.o.g. that the queries are unique; otherwise the simulator simply returns the same output on the same input without incrementing the query counter Q.)

Let $i = |\mathsf{id}|$ be the depth of id. If this is query number Q_i^* (i.e. $Q = Q_i^*$), define $H(\mathsf{id}) \leftarrow R_i^*$ and return $H(\mathsf{id})$.

Otherwise, if $Q \neq Q_i^*$:

1. Compute $A_i = A \cdot \big(R_{i-1}^* \cdots R_2^* R_1^*\big)^{-1} \in \mathbb{Z}_q^{m \times m}$ (where $A_1 = A$).
2. Run $\mathsf{SampleRwithBasis}(A_i)$ to obtain a random $R \sim \mathcal{D}_{m \times m}$ and a short basis T_B for $B = A_i R^{-1} \bmod q$.
3. Save the tuple $(i, \mathsf{id}, R, B, T_B)$ for future use, and return $H(\mathsf{id}) \leftarrow R$.

Secret key queries. $\mathcal{A}$ makes interactive key-extraction queries on arbitrary identities id, chosen adaptively. $\mathcal{B}$ answers a query on $\mathsf{id} = (\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_k)$ of length $|\mathsf{id}| = k \in [d]$ as follows.

1. Let $j \in [k]$ be the shallowest level at which $H(\mathsf{id}_{|j}) \neq R_j^*$. In the unlikely event that $H(\mathsf{id}_{|j}) = R_j^*$ for all $j = 1, \ldots, k$ the simulator aborts and fails.
2. Retrieve the saved tuple $(j, \mathsf{id}_{|j}, R, B, T_B)$ from the hash oracle query history. This tuple was created when responding to a query for $H(\mathsf{id}_{|j})$ (w.l.o.g., we can assume that an extraction query on id is preceded by a hash query on all prefixes of id). By construction
$$B = A \cdot (R_1^*)^{-1} \cdots (R_{j-1}^*)^{-1} \cdot H(\mathsf{id}_{|j})^{-1} \bmod q$$
and T_B is a short basis for $\Lambda_q^{\perp}(B)$.
Notice that B is exactly the encryption matrix $F_{\mathsf{id}_{|j}}$ (as defined in the Encrypt algorithm) for the ancestor identity $\mathsf{id}_{|j} = (\mathsf{id}_1, \mathsf{id}_2, \ldots, \mathsf{id}_j)$ and therefore T_B is a trapdoor for $\Lambda_q^{\perp}(F_{\mathsf{id}_{|j}})$.
3. Run $\mathsf{Derive}(\mathsf{PP}, T_B, \mathsf{id})$ to generate a secret key for id from the private key T_B for the identity $\mathsf{id}_{|j}$. Send the resulting secret key to the adversary.

Challenge. $\mathcal{A}$ announces to $\mathcal{B}$ the identity id^* on which it wishes to be challenged and a message $b^* \in \{0, 1\}$ to be encrypted. We require that id^* not be equal to, or a descendant of, any identity id for which a private key has been or will be requested in any preceding and subsequent key extraction query.

Let $\ell = |\mathsf{id}^*|$. If there is an $i \in [\ell]$ such that $H(\mathsf{id}^*_{|i}) \neq R_i^*$, then the simulator must abort. (Indeed, when this is the case, $\mathcal{B}$ is able extract a private key for id^* and thus answer by itself the challenge that it intended to ask.)

Recall that $A = A_0 R_w^* \cdots R_1^*$. If $w \neq \ell$ then the simulator aborts and fails.

Now, suppose $w = \ell$ and id^* is such that $H(\mathsf{id}^*_{|i}) = R_i^*$ for all $i \in [\ell]$. Then by definition
$$F_{\mathsf{id}^*} = A\,(R_1^*)^{-1} \cdots (R_\ell^*)^{-1} = A_0 \quad \in \mathbb{Z}_q^{n \times m}$$
and $\mathcal{B}$ proceeds as follows:

1. Retrieve $v_0, \ldots, v_m \in \mathbb{Z}_q$ from the LWE instance and set $v^* = \begin{bmatrix} v_1 \\ \vdots \\ v_m \end{bmatrix} \in \mathbb{Z}_q^m$.
2. Blind the message bit by letting $c_0^* = v_0 + b^* \lfloor \frac{q}{2} \rceil \in \mathbb{Z}_q$.
3. Set $c_1^* = v^* \in \mathbb{Z}_q^m$.
4. Set $\mathsf{CT}^* = (c_0^*, c_1^*)$ and send it to the adversary.

When $\mathcal{O}$ is a pseudo-random LWE oracle then $c_0 = u_0^\top s + x + b\lfloor \frac{q}{2} \rceil$ and $c_1 = F_{\mathsf{id}^*}^\top s + y$ for some random $s \in \mathbb{Z}_q^n$ and noise values x and y. In this case (c_0, c_1) is a valid encryption of b for id^*.

When $\mathcal{O}$ is a random oracle then (v_0, v^*) are uniform in $(\mathbb{Z}_q \times \mathbb{Z}_q^m)$ and therefore (c_0, c_1) is uniform in $(\mathbb{Z}_q \times \mathbb{Z}_q^m)$.

Now, $\mathcal{A}$ makes more secret key queries, answered by $\mathcal{B}$ in the same manner as before. Finally, $\mathcal{A}$ guesses whether CT^* was an encryption of b^* for id^*. $\mathcal{B}$ outputs $\mathcal{A}$'s guess and ends the simulation.

The distribution of the public parameters is identical to its distribution in the real system as are responses to private key queries. By Theorem 3, responses to H oracle queries are as in the real system. Finally, if $\mathcal{B}$ does not abort then the challenge ciphertext is distributed either as in the real system or is independently random in $(\mathbb{Z}_q, \mathbb{Z}_q^m)$. Hence, if $\mathcal{B}$ does not abort then its advantage in solving LWE is the same as $\mathcal{A}$'s advantage in attacking the system.

Since $\mathcal{A}$ is PPT it only finds collisions on H with negligible probability. A standard argument shows that the simulator can proceed without aborting with probability $\Pr[\neg\text{abort}] \geq Q_H^{-\ell}/d - \text{negl}(n) \geq Q_H^{-d}/d - \text{negl}(n)$ for some constant $c > 0$. Then if $\mathcal{A}$ has advantage $\epsilon \geq 0$, $\mathcal{B}$ has advantage at least $[\epsilon/(dQ_H^d)] - \text{negl}(n)$ in deciding the LWE problem instance.

5 Selectively Secure HIBE in the Standard Model

We briefly describe an HIBE of depth d that is selectively secure without random oracles. The details are in the full version of the paper. The construction is a binary tree encryption (BTE) which means that identities at each level are binary (i.e. 0 or 1). To build an HIBE with k-bit identities at each level we assign k levels of the BTE hierarchy to each level of the HIBE. The parameters used by this system are shown in Table 1.

Setup: For a BTE of depth d the setup algorithm runs $\mathsf{TrapGen}(q, n)$ to generate a random $n \times m$ matrix $A \in \mathbb{Z}_q^{n \times m}$ with a short basis $T_A \in \mathbb{Z}^{m \times m}$ for $\Lambda_q^\perp(A)$ and samples $2d$ matrices $R_{1,0}, R_{1,1}, \ldots, R_{d,0}, R_{d,1} \in \mathbb{Z}^{m \times m}$ from the distribution $\mathcal{D}_{m \times m}$ using $\mathsf{SampleR}(1^m)$. With u_0 random in $\mathbb{Z}_q^n$ the public params and master key are

$$\mathsf{PP} = \left(A\,,\ u_0\,,\ R_{1,0}, R_{1,1}\,,\ R_{2,0}, R_{2,1}\,,\ \ldots\,,\ R_{d,0}, R_{d,1} \right) \quad , \quad \mathsf{MK} = \left(T_A \right)$$

Extract: the secret key for an identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell) \in \{0,1\}^{\ell \leq d}$ is a short random basis for the lattice $\Lambda_q^\perp(F_{\mathsf{id}})$ where

$$F_{\mathsf{id}} = A\,(R_{1,\mathsf{id}_1})^{-1}\,(R_{2,\mathsf{id}_2})^{-1} \cdots (R_{\ell,\mathsf{id}_\ell})^{-1} \in \mathbb{Z}_q^{n \times m} \tag{5}$$

Encryption and decryption are as in the system of Section 4.1 using the matrix F_{id} from (5) in a dual-Regev encryption.

Security. The simulator is given an identity $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell) \in \{0,1\}^\ell$ where the attacker will be challenged. To simplify the description assume id is at maximum depth, namely $\ell = d$. The case $\ell < d$ is just as easy, but complicates the notation.

The simulator first constructs a matrix $A_0 \in \mathbb{Z}_q^{n \times m}$ from the given LWE challenge. It then samples random matrices

$$R_{1,\mathsf{id}_1}, R_{2,\mathsf{id}_2}, \ldots, R_{\ell,\mathsf{id}_\ell} \in \mathbb{Z}^{m \times m}$$

from the distribution $\mathcal{D}_{m \times m}$ and sets $A = A_0\ R_{\ell,\mathsf{id}_\ell} \cdots R_{2,\mathsf{id}_2}\, R_{1,\mathsf{id}_1} \in \mathbb{Z}_q^{n \times m}$. Now, consider the d matrices

$$F_i = A\,(R_{1,\mathsf{id}_1})^{-1} \cdots (R_{i,\mathsf{id}_i})^{-1} \quad \text{for } i = 0, \ldots, d-1.$$

For each matrix F_i the simulator invokes $\mathsf{SampleRwithBasis}(F_i)$ to obtain a matrix $R_{i,1-\mathsf{id}_i} \in \mathbb{Z}^{m\times m}$ and a short basis T_i for $\Lambda_q^{\perp}(F_i \cdot (R_{i,1-\mathsf{id}_i})^{-1})$. Finally, it sends to the adversary the public parameters

$$\mathsf{PP} \;=\; \Big(A\ ,\ \ u_0\ ,\ \ R_{1,0}, R_{1,1}\ ,\ R_{2,0}, R_{2,1}\ ,\ \ \ldots\ ,\ R_{d,0}, R_{d,1} \Big)$$

where u_0 is a random vector in $\mathbb{Z}_q^n$ from the LWE challenge.

It is not difficult to see that the simulator can use $T_1, \ldots, T_d$ to generate private keys for every node in the hierarchy except for the challenge identity id. Moreover, for the challenge identity it can generate a ciphertext that will help it solve the given LWE challenge as in Section 4.3, as required.

6 Conclusions

We presented a new lattice basis delegation mechanism and used it to construct two HIBE systems, one secure in the random oracle model and one secure without random oracles. The random oracle construction provides a lattice HIBE with short ciphertexts and private keys. The standard model system is not as short.

This work raises a number of interesting open problems. First, our standard model system processes bits of the identity one at a time. It would be interesting to apply the techniques of [ABB10, Boy10] to obtain a selective HIBE that processes many bits at a time so that the encryption matrix F_{id} is a product of only ℓ low-norm matrices for identities at depth ℓ.

Another interesting problem is an adaptively secure HIBE in the standard model where performance does not degrade exponentially in the hierarchy depth. Using the lattice basis delegation method from this paper or from [CHKP10] in Waters' dual encryption system [Wat09] is a promising direction.

Acknowledgments. We thank David Freeman, Daniele Micciancio and Brent Waters for helpful comments about this work.

References

[ABB10] Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)

[Ajt99] Ajtai, M.: Generating hard instances of the short basis problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)

[AP09] Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS, pp. 75–86 (2009)

[BB04] Boneh, D., Boyen, X.: Efficient selective-id secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)

[BBG05] Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)

[BF01] Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)

[Boy10] Boyen, X.: Lattices mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 499–517. Springer, Heidelberg (2010)

[BW06] Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)

[CDMP05] Coron, J.-S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-damgard revisited: how to construct a hash function. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)

[CHK07] Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. J. Crypto 20(3), 265–294 (2007); Abstract in Eurocrypt 2003 (2003)

[CHKP10] Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)

[GH09] Gentry, C., Halevi, S.: Hierarchical identity based encryption with polynomially many levels. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 437–456. Springer, Heidelberg (2009)

[GN08] Gama, N., Nguyen, P.: Predicting lattice reduction. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008)

[GPV08] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC (2008)

[GS02] Gentry, C., Silverberg, A.: Hierarchical id-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)

[HL02] Horwitz, J., Lynn, B.: Toward hierarchical identity-based encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 466–481. Springer, Heidelberg (2002)

[MG02] Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems: a cryptographic perspective, vol. 671. Kluwer Academic Publishers, Boston (March 2002)

[MR07] Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing (SICOMP) 37(1), 267–302 (2007); Extended abstract in FOCS 2004 (2004)

[Pei] Peikert, C.: Bonsai trees (or, arboriculture in lattice-based cryptography). Cryptology ePrint Archive, Report (2009), /359, `http://eprint.iacr.org/`

[Pei09] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: STOC, pp. 333–342 (2009)

[Reg09] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM 56(6) (2009); Extended abstract in STOC 2005 (2005)

[Sha85] Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)

[Wat05] Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)

[Wat09] Waters, B.: Dual key encryption: Realizing fully secure IBE and HIBE under simple assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009)