Weak is Better: Tightly Secure Short Signatures from Weak PRFs

Jacob Alperin-Sheriff[1]* and Daniel Apon[2]**

[1] National Institute of Standards and Technology,
jacob.alperin-sheriff@nist.gov
[2] University of Maryland,
dapon@cs.umd.edu

Abstract. The Boyen-Li signature scheme [Asiacrypt'16] is a major theoretical breakthrough. Via a clever homomorphic evaluation of a pseudorandom function over their verification key, they achieve a reduction loss in security linear in the underlying security parameter and entirely independent of the number of message queries made, while still maintaining short signatures (consisting of a single short lattice vector). All previous schemes with such an independent reduction loss in security required a linear number of such lattice vectors, and even in the classical world, the only schemes achieving short signatures relied on non-standard assumptions.

We improve on their result, providing a verification key smaller by a linear factor, a significantly tighter reduction with only a constant loss, and signing and verification algorithms that could plausibly run in about 1 second. Our main idea is to change the scheme in a manner that allows us to replace the pseudorandom function evaluation with an evaluation of a much more efficient weak pseudorandom function.

As a matter of independent interest, we give an improved method of randomized inversion of the $\mathbf{G}$ gadget matrix [MP12], which reduces the noise growth rate in homomorphic evaluations performed in a large number of lattice-based cryptographic schemes, without incurring the high cost of sampling discrete Gaussians.

* Most of this work was performed while the author was at the University of Maryland
** This work was performed under financial assistance award 70NANB15H328 from the U.S. Department of Commerce, National Institute of Standards and Technology.

1 Introduction

The Boyen-Li lattice-based signature scheme [BL16] from Asiacrypt 2016, is a theoretical breakthrough in terms of signature schemes with tight security reductions in the standard model.

Previous schemes achieving a reduction loss in security independent of the number of signing queries were either only proven secure in the random-oracle model [KW03, BLS04], required large signatures (of size at least quadratic in the security parameter) [BKKP15], or are insecure against quantum attacks [CW13, GHR99, BB08]. Their scheme manages to achieve short signatures (consisting of a single lattice vector in $\mathbb{Z}_q^m$) in the standard model with a loss in security depending (up to a constant factor) only on the loss from the security reduction of the pseudorandom function family (PRF) which underlies their scheme.

The starting point of their scheme is the Katz-Wang signature scheme, proven secure in the random oracle model [KW03]. Viewed at a high level, this scheme requires a pseudorandom function (PRF) PRF_k, a hash function H modeled as a random oracle, and a trapdoor function f. The signer computes a signature σ by inverting the trapdoor at $H(\mathsf{PRF}_k(\mu)||\mu)$, and the verifier accepts if $f(\sigma) = H(b||\mu)$ for some $b \in \{0,1\}$. In the security proof, the random oracle is programmed so that it can only invert the trapdoor function f at $f^{-1}(H(\mathsf{PRF}_k(\mu)||\mu))$, while learning $f^{-1}((H(1 - \mathsf{PRF}_k(\mu))||\mu))$ would allow it to solve some hard problem. If the PRF is secure, the latter case will happen with probability 1/2.

Boyen and Li notice that instead of using a random oracle, they can use the key-homomorphic trapdoor functions of Boneh et al. to encode the bits of the PRF key in the verification key [BGG+14]. To sign a message μ, they simply use the properties of the key-homomorphic trapdoor functions to homomorphically evaluate the PRF on μ, setting up the public key so they can only invert the trapdoor at $b = \mathrm{PRF}_{k,\mu}$. This enables them to gain tight security (with loss 1/2) in the same manner as Katz and Wang, without having to resort to the random oracle heuristic.

Evaluation of non-trivial circuits over homomorphic trapdoor functions comes at a price. In particular, while the Boyen-Li scheme can be instantiated with AES or any other block cipher as the PRF, these appear to be less than ideal choices. The reason for this is because block ciphers such as AES appear to require relatively high circuit depth to evaluate homomorphically [GHS12b], which forces the scheme to be based on the hardness of approximating SIS to within a subexponentially large factor. Unlike SIS with polynomially large factors, this problem can be solved in subexponential time using the BKZ algorithm [SE94, CN11].

As an alternative to assuming the subexponential hardness of SIS, they recall that a function with a circuit in NC^1 can be evaluated using the standard lattice-based homomorphic trapdoor constructions with only polynomial growth in the size of the underlying trapdoor [BV14, GV15]. Those PRFs which are known to have an NC^1 circuit are mostly based on quantum insecure assumptions [NR99, NRR02].

The only potentially post-quantum PRF with an NC^1 circuit that we are aware of is the ring-LWE (RLWE) based PRF of Banerjee et al. [BPR12], and indeed, this PRF is used by Boyen and Li for the quantum-secure instantiation of their scheme. The PRF of Banerjee et al., along with the quantum-insecure PRFs referenced above [3], all have a linear security loss in their reduction to the underlying hard problem on which they are based. As a result, Boyen and Li are able to instantiate their signature scheme with an overall loss in security linear in the underlying security parameter, but independent of the number of queries made by the adversary, which is a first for standard model lattice-based signature schemes with short signatures.

For completeness, we note that their paper also constructs a fully-secure IBE scheme enjoying very similar properties, where it enjoys some additional relative advantage to other such lattice-based schemes, as all such schemes require very large public keys (albeit not as large as theirs). However, our focus in this work is on the signature scheme only, as our main techniques do not transfer to the IBE setting.

1.1 Improving the Boyen-Li Scheme

Scheme	Pub. Key $R_q^{1\times k}$ mat.	Signature R_q^k vec.	Reduct. loss	Assumption(s)
[CHKP12]	n	n	Q	$\mathsf{RSIS}(n^{3/2})$
[Boy10]	n	1	Q	$\mathsf{RSIS}(n^{7/2})$
[MP12]	n	1	Q	$\mathsf{RSIS}(n^{5/2})$
[BHJ+14]	1	d	$(Q^2/\epsilon)^c$	$\mathsf{RSIS}(n^{5/2})$
[DM14]	d	1	$(Q^2/\epsilon)^c$	$\mathsf{RSIS}(n^{7/2})$
[Alp15]	1	1	$(Q^2/\epsilon)^c$	$\mathsf{RSIS}(d^{2d}\cdot n^{11/2})$
[BL16]	$nw\log^2 n$	1	λ	$\mathsf{RSIS}(w^4n^{7/2}), \mathsf{RLWE}(wn^{w/2})$
This work	$n\log n$	1	1	$\mathsf{RSIS}(n^{7/2}), \mathsf{LWR}(n)$

To avoid clutter, we ignore constant parameters above. We write $\mathsf{SIS}(\cdot)$ to specify the SIS parameter, and $\mathsf{LWE}(\cdot), \mathsf{LWR}(\cdot)$ to specify the LWE noise ratio and LWR rounding ratio, respectively. For SIS parameters, we also ignore logarithmic parameters, i.e. we write the parameter in $\tilde{O}$ notation without the $\tilde{O}$. The comparison is in the ring setting because as written, some of these schemes are only realizable in the ring setting. For those schemes using confined guessing or variants thereof, d is a value satisfying $2Q^2/\epsilon < 2^{\lfloor c^d \rfloor}$ for a constant $c = 2$. For the Boyen-Li Scheme, w is a parameter representing the length of input messages, and $\delta > 1$ is such that the PRF of Banerjee et al. [BPR12] can be computed by an NC^1 circuit of depth $\delta \log w$.

Fig. 1. Comparison to other standard model lattice-based signature schemes in the ring setting

[3] While they mention a DDH-based PRF construction by Jager that achieves a poly-logarithmic loss in security, the paper containing it has since been withdrawn.

Our main result is an improved version of the Boyen-Li signature scheme.

Better Parameters and Runtime. First, we improve greatly on the size of the public key and the hardness assumptions required. While our public key is still large (compared to some other lattice-based schemes), it is nevertheless a significant improvement over that of Boyen and Li. Although they never explicitly state the size of the public key in their work, it is clear that they need to encode the secret key for the PRF Banerjee et al. in the public key [BPR12], and furthermore, that at least one matrix is required in the public key for each bit of the PRF secret key in order to homomorphically evaluate the PRF using the ideas of Brakerski and Vaikuntanathan discussed above. The $nw \log^2 n = O(n^2 \log^2 n)$ value we give in Figure 1.1 then follows from the conditions required of the modulus of the PRF scheme.

Furthermore, we contend that while our public key is indeed large compared to those in [DM14, Alp15], the reduction loss in security in those schemes is so great as to make their proofs of security somewhat vacuous, while our reduction loss in security is constant (6). For an adversary making 2^{40} queries and succeeding with advantage 2^{-40} (which would be considered a practical "break" of the scheme), the reduction would yield an attack on the underlying hard problems that succeeds with advantage only 2^{-280}, which is almost certainly too small to be considered practically meaningful. By contrast, an adversary making 2^{40} queries and succeeding with advantage 2^{-40} against our scheme would yield an attack against the underlying hard problems with advantage 2^{-43}, which is still quite meaningful. For the other known standard-model lattice-based signature schemes, our public key is bigger by at most a logarithmic factor.

We also gain significantly in terms of the size required for the modulus q (which itself affects the size of the public key). They need to set the modulus to be at least
$q = \omega(\lambda^{4(1+c)})$, where the given PRF can be evaluated in depth $d = c \log \lambda$, and it takes time $\Omega(\lambda^{2c+1})$ to evaluate the PRF homomorphically, even in the ring setting. The specific values of c for pseudorandom functions known to be in NC^1 [BPR12, BP14, NR04, DS15] are not explicitly investigated by Boyen and Li (or by the authors of the papers in which the PRFs appear). However, the paper by Banerjee et al describes three sequential steps of a multi-product of vectors, a discrete Fourier Transform, and a rounding step, each of which can be seen to require at least $\log n$ depth (since each step depends on all n input bits) [RT92], which suggests that c is at least 3.

Our scheme gains even further in terms of runtime, and we discuss this point further below in Section 1.2.

Tighter and Weaker Assumptions. In addition to a much smaller SIS parameter of $n^{7/2}$ versus what appears to be $O(n^{7/2}w^{12}) = O(\lambda^{31/2})$ in the Boyen-Li paper (based on the apparent depth of the underlying PRF), we avoid the need for relying on the very strong and somewhat questionably quantum-safe[4]

[4] While no attacks on RLWE itself are known for subexponential large error ratios,Cramer, Ducas and Wesolowski recently gave a quantum polynomial-time algo-

assumptions like the hardness of RLWE for subexponentially large error ratios. Instead, our additional assumption is the hardness of the Learning with Rounding problem (LWR) over *general lattices* with a small (linear) rounding ratio. LWR over general lattices remains very plausibly quantum-hard for even for subexponential rounding ratios, and for the rounding ratio we require, we would expect any efficient (quantum) attack on LWR to be adaptable into an efficient attack on essentially all lattice-based cryptography; see Section 2.4 for some further justification.

1.2 Our Techniques

Our starting point is an investigation of the minimal security properties the function homomorphically evaluated in the Boyen-Li scheme must satisfy in order that the entire signature scheme be secure, i.e. existentially unforgeable against adaptive chosen-message attacks (eu-acma). We note that the adversary has the ability to choose which messages will be (homomorphically) evaluated by the function when computing the signature by simply asking the signing oracle to sign those messages. As a result, in order for the function's output to remain unpredictable by any means more successful than a random guess, the function must indeed be a strong pseudorandom function.

However, we recall that by hashing the message with a chamelon hash function [KR00] before signing, we can essentially eliminate an adversary's ability to choose which messages will be signed. In more detail, it has been shown [ST01] that a signature scheme that is existentially unforgeable against an adversary who can only observe signatures for (uniformly) *random* messages, can be turned into an euf-acma secure signature scheme by applying a chameleon hash function a concatenation of the message with some auxiliary sampled randomness, signing the output of the chameleon hash function instead of directly signing the message, and including the auxiliary sampled randomness as part of the signature.

Weak Pseudorandom Functions. Once the ability of the adversary to choose messages has been eliminated and the adversary is limited to observing signature on messages sampled uniformly at random, we now see that the function being homomorphically evaluated need only be a *weak* pseudorandom function (W–PRFs) [DN02]. In contrast to strong pseudorandom functions, the output must only remain unpredictable against an adversary who can observe the output of the function on any polynomial number of messages chosen uniformly at *random*.

At a complexity theoretic level, there is very strong evidence that weak pseudorandom functions are much simpler to compute than strong PRFs. In particular, Razborov and Rudich [RR94] have shown that any candidate strong pseudorandom function family computable in the complexity class $\mathsf{AC}^0(\mathsf{MOD}_2)$ of

rithm for approximating the worst-case shortest vector on ideal lattices to within a factor of $\exp(\tilde{O}(\sqrt{n}))$ [CDW16], making the reduction from RLWE to worst-case ideal lattice problems essentially vacuous

polynomial-size (in parameter λ) constant-depth circuit families with unbounded fan-in $\mathsf{AND}, \mathsf{OR}$ and MOD_2 gates can be only superpolynomially hard. In particular, they can be attacked by an adversary of size h with advantage at least $1/h$ for $h = O(\exp(\mathrm{poly}(\log n))$. By contrast, there exist candidate weak pseudorandom functions (that are plausibly exponentially hard) in this complexity class [ABG+14].

Instead of using the candidate weak pseudorandom functions in $\mathsf{AC}^0(\mathsf{MOD}_2)$, we instead opt for using the Learning With Rounding ($\mathsf{LWR}_{n,q,2}$) function family, for modulus $q = 2^\ell$ [BPR12]. Each function in the family is indexed by a secret $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and on input $\mathbf{a} \in \mathbb{Z}_q^n$ outputs

$$f_{\mathbf{s}}(\mathbf{a}) = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rceil_2 = \lfloor \tfrac{2}{q} \langle \mathbf{a}, \mathbf{s} \rangle \rceil \bmod 2.$$

Under the assumption that $\mathsf{LWR}_{n,q,2}$'s output is indistinguishable from a truly random function (when evaluated on $\mathbf{a}$ sampled uniformly at random), known as the *decisional*-$\mathsf{LWR}_{n,q,2}$ assumption, we can immediately see that the LWR function is a pseudorandom function. It is also very easy to see that it is not a strong PRF, because one can learn the jth most significant bit of the ith coordinate of $\mathbf{s}$ by making a query on input $2^{j-1}\mathbf{e}_i$, where $\mathbf{e}_i$ is the ith vector of the standard basis.

Efficient Homomorphic Evaluation of LWR. The main reason we opt for using LWR as our weak PRF is that, viewed in the above manner, it is *identical* to the *decryption function* for most lattice-based encryption schemes [Reg09]. Homomorphic evaluation of the decryption function, of course, is better known in fully homomorphic encryption (FHE) contexts as bootstrapping, where it is a central operation necessary to allow unbounded homomorphic computation [Gen09]. A large body of work has focused on optimizing the evaluation of this function for various fully homomorphic encryption schemes [GHS12a, AP13, OvdPS15, HS15], but for our purposes, we are particularly interested in those works focused on bootstrapping LWE ciphertexts using the Gentry-Sahai-Waters (GSW) [GSW13] encryption scheme [ASP14, HAO15, DM15, CGGI16].

This interests stems from the extreme similiarities between homomorphic evaluations over GSW ciphertexts and homomorphic evaluations over key-homomorphic trapdoor functions. In particular, as has been noted implicitly by Gorbunov and Vinayagamurthy, the cost of a given sequence of homomorphic operations in terms of error growth in GSW ciphertexts is essentially identical to the cost of that same sequence of operations in terms of trapdoor growth over key-homomorphic trapdoor functions [GV15].

As a result, algorithms using the GSW encryption scheme to boostrap LWE can in fact be used in our context to homomorphically evaluate the LWR function over key-homomorphic trapdoor functions. While FHE has generally become known for its inefficiency, and it is indeed very very far away from being useful for its main intended use, a single bootstrapping operation has recently moved much closer to efficiency. An implementation by Ducas and Miccancio [DM15] in the ring setting was notably able to implement bootstrapping for plausible

security parameters in about 0.6 seconds. As homomorphic evaluation of the LWR function should be the most computationally intensive part of signing and verifying, this suggests plausibly being able to sign and verify in under a second.

We note that a more recent work by Chillotti et al. [CGGI16], to appear in Asiacrypt 2016, is faster than the work of Ducas and Micciancio for similar security parameters by a factor of 12, but does not, strictly speaking, perform its evaluation over full GSW ciphertexts, and hence we cannot reasonably extrapolate anything about the speed of our construction from it.

In addition to the more standard primitives of chameleon hash functions and weak pseudorandom functions (see below), we instantiate our signature scheme using Puncturable Homomorphic Trapdoor Functions (PHTDFs), which were introduced by Alperin-Sheriff [Alp15] as an extension of the Gorbunov et al. definition of homomorphic trapdoor functions [GVW15b] and the Boneh et al. definition of key-homomorphic trapdoor functions [BGG$^+$14]. PHTDFs give a formal way to encapsulate out repeatedly used properties in lattice-based signature schemes, and avoid having to repeat technical arguments in each new lattice-based signature schemes.

Reducing Trapdoor Growth Rates. In addition to the Boyen-Li scheme, key-homomorphic trapdoor functions [BGG$^+$14] have been at the heart of wide variety of cryptographic constructions in recent years, including attributed-based encryption and predicate encryption schemes [BP14, GVW15a, GVW15b]. In the standard lattice-based construction, a key operation is sampling a matrix $\mathbf{X} \in \mathbb{Z}_q^{m \times m}$ with small norm such that $\mathbf{GX} = \mathbf{U} \pmod q$ for the so-called "gadget" matrix $\mathbf{G}$, which we denote $\mathbf{G}^{-1}(\mathbf{U})$.

In all existing schemes using key-homomorphic trapdoor functions, $\mathbf{G}^{-1}$ is evaluated via deterministic bit decomposition. While efficient (taking linear time) and fully parallelizable, it has downsides when trying to limit the noise growth (or trapdoor growth), as each column can have length as large as m. More concretely, when $\mathbf{G}^{-1}$ is evaluated in this deterministic manner, for a (fixed) vector $\mathbf{a} \in \mathbb{Z}^m$ and random vector $\mathbf{u} \in \mathbb{Z}_q^n$, we expect that $\|\langle \mathbf{a}, \mathbf{G}^{-1}(\mathbf{u}) \rangle\| \approx \frac{m}{2}\|\mathbf{a}\|$.

However, $\mathbf{G}^{-1}$ can also be evaluated in a randomized manner (sampled). Indeed, Miccancio and Peikert first defined and analyzed the gadget matrix $\mathbf{G}$ [MP12] with discrete Gaussian sampling applications in mind. Using Gaussian sampling instead of deterministic bit decomposition to evaluate $\mathbf{G}$ can reduce noise growth from m to $\sqrt{m}$ [ASP14]. This ultimately allows for basing the security of the given scheme on a harder problem and allows for smaller parameters for real security against known attacks.

For several reasons, this randomization technique has not been used in schemes utilizing key-homomorphic trapdoor functions. First, in all key-homomorphic trapdoor function-based schemes, the function evaluated on the master public and secret key by the key-generating authority when generating a given secret key (or signature) must correspond to the function evaluated by a user holding that secret key (or signature) for a ciphertext (or a set of trapdoor functions) that the given secret key is authorized to decrypt (or that the given signature

should be verifiable on). In particular, it necessary that both the key-generating authority and the user perform each evaluation of $\mathbf{G}^{-1}$ in the same manner. If a randomized version of $\mathbf{G}^{-1}$ is used, the bits used by they key-generating authority in evaluating $\mathbf{G}^{-1}$ will have to be transferred in some manner to the user holding the secret key. As many as n bits are required in order for a single sample to be within $2^{-\Omega(n)}$ distance of the actual discrete Gaussian distribution when using rejection sampling [GPV08]. One can also use tables to help with the computation (which is advisable because rejection sampling does not run in so-called "constant time"), but table lookups come with their own set of disadvantages [DG14].

We present a much simpler randomized method of sampling $\mathbf{G}^{-1}$ which is nearly as fast as bit decomposition and does not require the use of tables. We also discuss the manner in which the randomness is transferred to the user. In some schemes, one is "stuck" with using the trivial method of including the randomness in each signature as a separate element. While this ultimately requires a signature with more elements on a conceptual level, the reduction in the size of the modulus q and dimension n required for an equivalent level of security that the randomized sampling allows us will more than make up for this addition, resulting in less actual bits required per signature to achieve a given security level.

However, in schemes which already use a chameleon hash function to randomize the input messages, including but not limited to [Alp15, MP12, Boy10, DM14], we show that we do not need to include any additional elements in the signature. Instead, it is sufficient to include the randomness in the scheme's public key, and for the signing algorithm to "reuse" it on all signatures it issues, without more than a negligible loss in effectiveness in reducing the rate of noise growth.

1.3 Open Problems

While it provides massive improvements in the runtime of the signature scheme and improves the size of the parameters somewhat over the Boyen-Li Scheme, our scheme still requires a very large public key. The "holy grail" of achieving a standard-model signature scheme with a tight reduction to a plausible hardness assumption, while keeping both signatures and public keys small, remains open.

2 Preliminaries

We write $[d]$ to denote the set of positive integers $\{1, \ldots, d\}$. For an integer $q \geq 2$, we use $\mathbb{Z}_q$ to denote the ring of integers modulo q, and somewhat abuse notation by also using $\mathbb{Z}_q$ to explicitly represent the integers in $(-q/2, q/2]$. We define $|x| \in \mathbb{Z}_q$ by taking the absolute value of the representative in this range.

We use $\otimes$ to denote the Kronecker product of two matrices.

2.1 Signatures

We briefly recall the standard definitions of digital signature schemes. A *signature scheme* SIG is a triple $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ of PPT (probabilistic polynomial time)

algorithms, together with a message space $\mathcal{M} = \mathcal{M}_\lambda$. It is correct if, for all messages $\mu \in \mathcal{M}_\lambda$, $\mathsf{Ver}(vk, \mu, \sigma) = 1$ holds true, except with negligible probability in λ over the choice of $(sk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{Sign}(sk, \mu)$.

We now recall the standard security definitions for digital signature schemes. Existential unforgeability under adaptive chosen-message attack, or eu-acma, is as follows: the adversary The challenger generates keys $(vk, sk) \leftarrow \mathsf{Gen}$ and sends vk to $\mathcal{A}$. At this point, $\mathcal{A}$ can adaptively requests signatures on messages μ, and the challenger responds with $\sigma = \mathsf{Sign}(sk, \mu)$, for each message.Finally, $\mathcal{A}$ outputs an attempted forged signature (μ^*, σ^*). In order to satisfy eu-acma security, the probability that $\mu^* \neq \mu_i$ for any $i \in [Q]$ and that $\mathsf{Ver}(vk, \mu^*, \sigma^*) = 1$ accepts should be negligible in the security parameter λ.

Chameleon Hashing Chameleon hash functions, invented by Krawczyk and Rabin [KR00], have been applied to signature schemes for a number of purposes, notably for generic transformations from statically-secure signatures (where the challenger receives the messages to be signed before giving the adversary the verification key) to adaptively-secure signatures (where the security game proceeds as above). As our usage of chameleon hash functions is slightly non-standard, we briefly recall their definitions, following the variant definition of Ducas and Micciancio specialized to their lattice-based construction [DM14].

Definition 2.1. *A chameleon hash function family is a set of three algorithms* $CH = (\mathsf{Gen}, \mathsf{Hash}, \mathsf{Hash}^{-1})$ *along with an efficiently computable input distribution* $\mathcal{X}_n$, $\mathcal{Y}_n$ *for each integer* n, *where* $\perp \notin \mathcal{Y}_n$. *Except with negligible probability over the choice of* $(ek, td) \leftarrow \mathsf{Gen}(1^n)$, *the following should hold for security.*

Uniformity: *For a fixed message* μ, *evaluation key* ek, *trapdoor* td, $(x \leftarrow \mathcal{X}_n, y \leftarrow \mathsf{Hash}_{ek}(\mu, x))$ *should be distributed within negligible statistical distance of* $(x \leftarrow \mathsf{Hash}_{td}^{-1}(\mu, y), y \leftarrow \mathcal{Y}_n)$

Collision Resistance: *Given only access to* ek *and public parameters, it should be hard for any PPT algorithm* $\mathcal{A}$ *to output* $(\mu, r) \neq (\mu', r')$ *such that* $\mathsf{Hash}_{ek}(\mu, r) = \mathsf{Hash}_{ek}(\mu', r') \neq \perp$.

We also note that the Ducas and Micciancio paper provides an explicit ring-based construction that is significantly more efficient in both space and time than our main construction, and that this construction has a very straightforward adaptation to general lattices and module lattices [LS15].

2.2 Lattices

The main result of the paper in Section 3 abstracts out the concrete lattice details in the form of Puncturable Homomorphic Trapdoor Functions (PHTDFs); see Section 2.5 below. However, for our complementary result on reducing the trapdoor growth for signature schemes (Section 4), we do need to recall some very basic results.

Following Gentry et al. [GPV08], for integers $n \geq 1$, modulus $q \geq 2$, we define the m-dimensional lattice specified by an "arity check" matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$:

$$\Lambda^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \in \mathbb{Z}_q^n\} \subseteq \mathbb{Z}^m,$$

and for $\mathbf{y}$ in the subgroup of $\mathbb{Z}_q^n$ generated by the columns of $\mathbf{A}$, we define the coset

$$\Lambda_{\mathbf{y}}^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{y} \bmod q\} = \Lambda^{\perp}(\mathbf{A}) + \bar{\mathbf{x}},$$

where $\bar{\mathbf{x}} \in \mathbb{Z}^m$ is an arbitrary solution (not necessarily short) to $\mathbf{A}\bar{\mathbf{x}} = \mathbf{y}$.

"Gadget" matrix $\mathbf{G}$ We recall the gadget matrix $\mathbf{G}$ defined by Micciancio and Peikert [MP12]. We focus on the case that the modulus $q = 2^k$ for ease of analysis.

We define

$$\mathbf{g}^t = [1, 2, 2^2, \ldots, 2^{k-1}] \in \mathbb{Z}_q^{1 \times k}$$

Then we have that

$$\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^t \in \mathbb{Z}_q^{n \times nk}$$

While $\mathbf{G}$ is used by Miccancio and Peikert to sample discrete Gaussians, in this work we only use it for computing a simpler distribution (see Section 4 for details).

The SIS problem. For $\beta > 0$, the *short integer solution* problem $\mathsf{SIS}_{n,q,\beta}$ is an average-case version of the approximate shortest vector problem on $\Lambda^{\perp}(\mathbf{A})$. Given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for any $m = \mathrm{poly}(n)$, the problem is to find a nonzero vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{z} = \mathbf{0} \bmod q$ and $\|\mathbf{z}\| \leq \beta$. For $q \geq \beta\sqrt{n}\omega(\sqrt{\log n})$, it has been shown that solving this problem with non-negligible success probability over the random choice of $\mathbf{A}$ is at least as hard as probabilistically approximating the classic Shortest Independent Vectors Problem (SIVP) on n-dimensional lattices to within $\tilde{O}(\beta\sqrt{n})$ factors in the *worst case*. [Ajt04, MR07, GPV08]. Analogous worst-case reductions exist for the more general case of module lattices, where $\mathbf{A} \in R_q^{d \times m}$ for some arbitrary ring of integers R algebraic number field K [LS15], which essentially includes the above results as a special case.

2.3 Subgaussian Random Variables

To analyze our distribution in Section 4, we make use of the notion of *subgaussian* random variables. (For further details and full proofs, see [Ver12].) A random vector $\mathbf{x}$ is subgaussian with parameter $r > 0$ if for all $t \in \mathbb{R}$ and all (fixed) real unit vectors $\mathbf{u}$, its (scaled) moment-generating function satisfies $\mathbb{E}[\exp(\langle \mathbf{u}, \mathbf{x} \rangle)] \leq \exp(Cr^2t^2)$ for an absolute constant C (for our application, we may take $C = 1$). By a Markov argument, for all $t \geq 0$, we have

$$\Pr[\|\mathbf{x}\| \geq t] \leq 2\exp(-t^2/r^2). \tag{1}$$

Any B-bounded centered random vector $\mathbf{x}$ (i.e., $\mathbb{E}[\mathbf{x}] = \mathbf{0}$ and $|X| \leq B$ always) is subgaussian with parameter B.

We recall the following additional properties of subgaussian vectors $\mathbf{x}$ [Ver12].

Homogeneity: If $\mathbf{x}$ is subgaussian with parameter r, then $c\mathbf{x}$ is subgaussian with parameter $c \cdot r$ for any constant $c \geq 0$.

Pythagorean additivity: if $\mathbf{x}_1$ is subgaussian with parameter r_1, and $\mathbf{x}_2$ is subgaussian with parameter r_2 conditioned on *any* value of $\mathbf{x}_1$ (in particular, if $\mathbf{x}_2$ is subgaussian with parameter r_2 and independent of $\mathbf{x}_1$), then $\mathbf{x}_1 + \mathbf{x}_2$ is subgaussian with parameter $\sqrt{r_1^2 + r_2^2}$.

Euclidean Norm : Let $\mathbf{x} \in \mathbb{R}^n$ be a random vector with independent coordinates that are subgaussian with parameter r. Then for some (small) universal constant $0 < C$, we have $\Pr[\|\mathbf{x}\|_2 > C \cdot r\sqrt{n}] \leq 2^{-\Omega(n)}$

2.4 Weak Pseudorandom Functions and Learning with Rounding

Here we give a basic definition of weak pseudorandom functions [DN02].

More formally, a weak pseudorandom function family (outputting a single bit) $\mathsf{W\text{–}PRF} : \{0,1\}^\lambda \times \{0,1\}^m \to \{0,1\}$ is considered secure if no probabilistic polynomial-time adversary can distinguish a member of the family $f_{\mathbf{k}} : \{0,1\}^m \to \{0,1\}, f_{\mathbf{k}} := \mathsf{W\text{–}PRF}(\mathbf{k}, \cdot)$ (where $\mathbf{k} \leftarrow \{0,1\}^\lambda$ uniformly at random) from a truly random function with advantage greater than $\mathrm{negl}(\lambda)$, given that it can observe

$$(x_1, f_{\mathbf{k}}(x_1)), \ldots, (x_m, f_{\mathbf{k}}(x_m))$$

for any $m \in \mathrm{poly}(\lambda)$, where each $x_1, \ldots, x_m$ is sampled uniformly at random from $\{0,1\}^m$.

A concrete candidate weak pseudorandom function family is the learning with rounding function family ($\mathsf{LWR}_{n,Q,p}$) [BPR12]. Functions in the family are indexed by a secret key $\mathbf{s} \in \mathbb{Z}_Q^n$. For a given secret key $\mathbf{s}$, the function is defined as

$$\mathsf{LWR}_{n,Q,p}(\mathbf{a}) = \lfloor \tfrac{p}{Q} \langle \mathbf{a}, \mathbf{s} \rangle \rceil,$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer.

LWR has been shown to be a weak pseudorandom function under the better-known LWE assumption with discrete Gaussian noise terms (and hence on worst-case shortest vector problems on lattices) in a number of different results [BPR12, AKPW13, ASA16, BGM+16, BLL+15]. While all of these results require the ratio Q/p to grow with the number of samples revealed (meaning that hardness for non-a-priori bounded $m = \mathrm{poly}(\lambda)$ requires assuming the shortest vector problem is hard to approximate to within a superpolynomial ratio), there is a reduction [BGM+16] with a sample loss ratio of Q/p in security from LWE with bounded uniform error to LWR. This latter result strongly suggests that LWR remains a weak pseudorandom function for any $Q/p = \Omega(\sqrt{n})$, and that the weaker reductions from LWE with Gaussian error are likely all artifacts of the proof techniques used.

2.5 Puncturable Homomorphic Trapdoor Functions

We recall Alperin-Sheriff's definition of Puncturable Homomorphic Trapdoor Functions (PHTDFs).

$pk \leftarrow \mathsf{Gen}(1^\lambda)$ takes as input a security parameter λ, which for a concrete instantiation implictly defines parameters for a ring $\mathcal{T}$ representing a tag space, a trapdoor space $\mathcal{R}$, a tagged function space $\mathcal{A}$, an index space $\mathcal{X}$, an input space $\mathcal{U}$ and an output space $\mathcal{V}$, and then generates the public key for the PHTDF. $\mathcal{R}$ and $\mathcal{U}$ are associated with parameterized efficiently sampleable distributions $D_{\mathcal{R},\beta}, D_{\mathcal{U},s}$, with the distribution details depending on the instantiation.

$(a, r) \leftarrow \mathsf{GenTrap}(pk, t)$ generates a *trapdoor* $r \leftarrow D_{\mathcal{R}}$ for the (pk, a), with t the tag *associated with* a, r. We need the distribution of a to be statistically close to uniform over $\mathcal{A}$.

$t \leftarrow \mathsf{Tag}(pk, a, r)$ is an auxiliary function which outputs the tag t associated with a and r is a trapdoor for (pk, a).

$f_{pk,a,x} : \mathcal{U} \to \mathcal{V}$ is a deterministic function indexed by $pk, x \in \mathcal{X}, a \in \mathcal{A}$.

$\mathsf{Invert}_{r,pk,a,x,s} : \mathcal{V} \to \mathcal{U}$ is a trapdoor-inverter indexed by $x \in \mathcal{X}$, $r \in \mathcal{R}$ and $pk, a \in \mathcal{A}$. If $f_{pk,a,x}$ is not injective, then Invert is a probabilistic function, and the parameter $s \in \mathbb{R}$ relates to the noise level $\mathsf{Prop}(u)$ of the inverse u output by Invert; in particular, we want $\mathsf{Prop}(u) \leq s$. We require that $\mathsf{Prop}(r) = \beta$ should be small enough to allow Invert to invert with parameter s when the tag t associated with a, r is *invertible* over the ring $\mathcal{T}$. If t is not invertible, then the trapdoor is considered *punctured*, and Invert outputs $\perp$.

$r^* \leftarrow \mathsf{Eval}^{td}_{pk}(g, \{(a_i, r_i)\}_{i\in[\kappa]}, \mathbf{y}), a^* \leftarrow \mathsf{Eval}^{func}_{pk}(g, \{a_i\}_{i\in[\kappa]}, \mathbf{y})$ are deterministic trapdoor/function homomorphic evaluation algorithms, respectively. The algorithms take as input some function $g : \mathcal{T}^\kappa \times \mathcal{T}^w \to \mathcal{T}$, a vector $\mathbf{y} \in \mathcal{T}^w$, as well as functions $a_i \in \mathcal{A}$ with associated trapdoors $r_i \in \mathcal{R}$. The outputs are $r^* \in \mathcal{R}$ and $a^* \in \mathcal{A}$.

Let $\mathbf{t} \in \mathcal{T}^\kappa$ be a vector such that t_i is the tag associated with a_i, r_i. We refer to g as *admissible* with parameter s on $\mathbf{t}$ if for any $v \in \mathcal{V}, \mathbf{y} \in \mathcal{T}^w$ such that $g(\mathbf{t}, \mathbf{y})$ is invertible, $\mathsf{Invert}_{r^*,pk,a^*,x,s}(v)$ successfully outputs u such that with overwhelming probability, $\mathsf{Prop}(u) \leq s$.

Security Properties. The following should hold for $pk \leftarrow \mathsf{Gen}(1^\lambda)$, trapdoor and function pair (r, a) with an invertible tag t:

$$(pk, r, a, x, u, v) \overset{s}{\approx} (pk, r, a, x, u', v')$$

where $x \in \mathcal{X}$ is arbitrary, $u \leftarrow D_{\mathcal{U},s}$, $v := f_{pk,a,x}(u)$, $v' \leftarrow \mathcal{V}$ and $u' \leftarrow \mathsf{Invert}_{r,pk,a,x,s}(v')$.

The security game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ is parameterized by a security parameter λ, as well as a function $g : \mathcal{T}^\kappa \times \mathcal{T}^w \to \mathcal{T}$ such that g is admissible with some parameter s on some subset of tags $\mathcal{S} \subseteq \mathcal{T}^\kappa$

1. $\mathcal{C}$ runs $pk \leftarrow \mathsf{Gen}(1^\lambda)$ and then computes $(a_i, r_i) \leftarrow \mathsf{GenTrap}(pk, \mathbf{t})$ for each $i \in [\kappa]$. $\mathcal{A}$ is given pk and $\{a_i\}$.
2. $\mathcal{A}$ may make (a polynomial number of) inversion queries, sending some $v \in \mathcal{V}$, $x \in \mathcal{X}$ and some $\mathbf{y} \in \mathcal{T}^w$ such that $g(\mathbf{s}, \mathbf{y})$ is *invertible*. $\mathcal{C}$ computes $r' \leftarrow \mathsf{Eval}^{td}_{pk}(g, \{(a_i, r_i)\}, \mathbf{y})$ as well as $a' \leftarrow \mathsf{Eval}^{func}_{pk}(g, \{a_i\}, \mathbf{y})$, samples $u \leftarrow \mathsf{Invert}_{r',pk,a',x,s}(v)$ and returns u to $\mathcal{A}$.

3. $\mathcal{A}(1^\lambda)$ outputs tag sets $\mathbf{y}^{(1)}, \mathbf{y}^{(2)} \in \mathcal{T}^w$ which satisfy $g(\mathbf{t}, \mathbf{y}) = g(\mathbf{t}, \mathbf{y}') = 0$, as well as $u^{(1)}, u^{(2)}, x^{(1)}) \neq x^{(2)}$, and wins if

$$f_{pk,a^{(1)},x^{(1)}}(u^{(1)}) = f_{pk,a^{(2)},x^{(2)}}(u^{(2)}),$$

where $\mathsf{Prop}(u^{(1)}), \mathsf{Prop}(u^{(2)}), \mathsf{Prop}(x^{(1)}), \mathsf{Prop}(x^{(2)}) \leq s$ and $a^{(b)} \leftarrow \mathsf{Eval}_{pk}^{func}(g, \{(a_i)\}, \mathbf{y}^{(b)})$ for $b \in \{1, 2\}$.

We say the PHTDF satisfies $(\epsilon = \epsilon(\lambda), t = t(\lambda), g, \mathcal{S})$-collision resistance when punctured (CRP) security if every PPT adversary taking at most time t has success probability at most ϵ in this game.

Concrete Instantiation The instantiation in [Alp15] is written in terms of general lattices, but as mentioned in that work, can easily be instantiated over rings or modules [LS15]. We briefly recall the relevant results on security, leaving tag instantiation descriptions to later in the paper.

Theorem 2.2 ([Alp15]). *Let g be admissible with parameter s. If there exists an adversary $\mathcal{A}$ breaking $CRP_{\epsilon,t,g,\mathcal{S}}$ security of the PHTDF, then there exists $\mathcal{A}'$ running in time t that solves $\mathsf{SIS}_{n,q,\beta}$ with advantage $\epsilon - \mathrm{negl}(\lambda)$ for $\beta = O(s^2\sqrt{n \log q})$.*

We will also need to recall the growth rate of the $\mathsf{Prop}(r)$ for the trapdoors used in this instantiation as a result of homomorphic operations. In particular, we have

Homomorphic addition of a_1 and a_2 with trapdoors r_1, r_2 induces a new trapdoor r^* with $\mathsf{Prop}(r^*) = \mathsf{Prop}(r_1) + \mathsf{Prop}(r_2)$

Homomorphic Multiplication of a_1, a_2 with trapdoor r_1, r_2, tags t_1, t_2 induces trapdoor r^* with

$$\mathsf{Prop}(r^*) = \mathsf{Prop}(r_1)\mathbf{G}^{-1}(a_2) + \mathsf{Prop}(t_1)\mathsf{Prop}(r_2)$$

A key trick with homomorphic multiplication is to chain them together in a left-associative manner, causing a quasi-additive growth in the trapdoors [BV14, ASP14].

3 Improved Signature Scheme With Tight Security

In this section, we define our improved signature scheme, and prove security with a tight reduction. We also discuss how to homomorphically evaluate our weak PRF in an efficient manner.

To avoid the clunky notation that has characterized many lattice-based constructions (and indeed, many cryptographic constructions in general), we write the scheme generically in terms of the PHTDF cryptographic primitive, as well as in terms of a generic weak PRF and chameleon hash function, and then focus on the homomorphic evaluation details in Section 3.4.

3.1 Parameters

Here we discuss the basic parameters of the scheme and the requirements they need to satisfy. All parameters in the scheme are parameterized by an underlying security λ. We have that $\alpha, w = \theta(\lambda)$. We will have that $\mathcal{T} = \mathbb{Z}_q$. The form of the parameter $\kappa \geq 2$ depends on the concrete instantiation and evaluation of the PHTDF. We defer details to Section 3.4.

We need $\mathsf{W\text{–}PRF} : \mathcal{T}^{\kappa-2} \times \mathcal{T}^{w-1} \to \{0,1\}$ be an $(\epsilon_{\mathsf{W\text{–}PRF}}, t_{\mathsf{W\text{–}PRF}})$-secure weak pseudorandom function, where $\mathcal{T}^{\kappa-2}$ is the keyspace and $\mathcal{T}^{w-1}$ is the input space.

For convenience reasons, we also need the PRF to be trivial in the (negligibly likely) case that $\mathcal{T}^{k-2} = \mathbf{0}$; formally, we need $\mathsf{W\text{–}PRF}(\mathbf{0}, \cdot) = 0$ for all $\mathbf{y} \in \mathcal{T}^{w-1}$. This property is satisfied naturally by LWR, and it and we can construct such a weak PRF from an arbitrary weak PRF $\mathsf{W\text{–}PRF}'$ by setting

$$\mathsf{W\text{–}PRF}(\mathbf{s}, \mathbf{y}) := \mathsf{W\text{–}PRF}'(\mathbf{s}, \mathbf{y}) \oplus \mathsf{W\text{–}PRF}'(\mathbf{0}, \mathbf{y}).$$

We also need $\mathrm{CH} = (\mathsf{Gen}, \mathsf{Hash}, \mathsf{Hash}^{-1})$ to be a secure chameleon hash function family such that the input space $\mathcal{X}$ is efficiently sampleable and grows in size as a (polynomial) function inthe underlying security parameter λ and such that the output space $\mathcal{Y} = \mathcal{T}^{w-1}$. This can be achieved essentially without loss of generality, as long as there is an efficiently computable embedding of the output space $\mathcal{Y}$ into $\mathcal{T}^{w-1}$. We use $w-1$ instead w because we will be appending a bit b to $\mathbf{y} \in \mathcal{T}^{w-1}$ as input to the function g that will be evaluated homorphically.

We define $g : \mathcal{T}^{\kappa} \times \mathcal{T}^{w} \to \mathcal{T}$ as

$$g((\mathbf{s}, z^{(0)}, z^{(1)}), (\mathbf{y}, b)) = (1 - z^{(b)}) - \mathsf{W\text{–}PRF}(\mathbf{s}, \mathbf{y}). \tag{2}$$

In the above equation, $\mathbf{y} \in \mathcal{T}^{w-1}$ will be the output of a chameleon hash function, and $\mathbf{s} \in \mathcal{T}^{\kappa-2}$ will be the secret key for the weak PRF.

Finally, we need $\mathrm{PHT} = (\mathrm{PHT}.\mathsf{Gen}, \mathrm{PHT}.\mathsf{GenTrap},$
$\mathrm{PHT}.f, \mathrm{PHT}.\mathsf{Invert}, \mathrm{PHT}.\mathsf{Eval}_{pk}^{td}, \mathrm{PHT}.\mathsf{Eval}_{pk}^{func})$ to be an $(\epsilon_{\mathrm{PHT}}, t_{\mathrm{PHT}}, g, \mathcal{S})$ CRP-secure PHTDF family, where $\mathcal{S} = \mathcal{T}^{\kappa-2}$.

3.2 Construction

We explicitly sample the tag encoded in the public key and add it to the signing key for use in signing. We also make the use of a chameleon hash function explicit, and we implicitly require the input space $\mathcal{X}$ for the chameleon hash function to satisfy shortness properties (and be such that $|\mathcal{X}| = 2^{\Omega(\lambda)}$, which is indeed the case for the Ducas-Micciancio chameleon hash function family construction.

Encoding the PRF secret key. Often, including in the instantiation we detail below in Section 3.4, the way of encoding the weak PRF secret key $\mathbf{s}$ into the verification key will be something other than its "natural representation," and we abstract this out with the additional utility function $\mathsf{Encode} : \mathcal{T}^{\kappa} \to \mathcal{T}^{\ell}$.

$\mathsf{Gen}(1^{\lambda})$ On input security parameter λ,

1. Choose $v \leftarrow \mathcal{V}$, $\mathbf{s} \leftarrow \mathcal{T}^{\kappa-2}$. Set $z^{(0)} = 0, z^{(1)} = 1$. Compute $\tilde{\mathbf{z}} = \mathsf{Encode}(\mathbf{s}, z^{(0)}, z^{(1)}) \in \mathcal{T}^{\ell}$.
2. Compute $(ek, td) \leftarrow \text{CH.}\mathsf{Gen}(1^{\lambda})$, $pk \leftarrow \text{PHT.}\mathsf{Gen}(1^{\lambda})$
3. Compute $\{(a_i, r_i) \leftarrow \text{PHT.}\mathsf{GenTrap}(pk, \tilde{\mathbf{z}}_i)\}_{i \in [\ell]}$.
4. Output $vk = (pk, \mathbf{a} = \{a_i\}_{i \in [\ell]}, ek, v)$, $sk = (\mathbf{r} = \{r_i\}_{i \in [\ell]}, td, \mathbf{z})$

$\mathsf{Sign}(vk, sk, \mu)$ On input message $\mu \in \{0,1\}^{\alpha}$, secret key sk, verification key vk:

1. Sample the input randomness to the chameleon hash function $x \leftarrow \mathcal{X}$, compute the output $\mathbf{y} = \text{CH.}\mathsf{Hash}(x, \mu)$, and then evaluate the weak pseudorandom function at $\mathbf{y}$ to get $b = \mathsf{W\text{–}PRF}(\mathbf{s}, \mathbf{y})$.
2. Perform the homomorphic evaluation of g over the PHTDF to get $r^* \leftarrow \mathsf{Eval}_{pk}^{td}(g, (\mathbf{a}, \mathbf{r}), (\mathbf{y}, b))$, $a^* \leftarrow \mathsf{Eval}_{pk}^{func}(g, \mathbf{a}, (\mathbf{y}, b))$.
3. Invert the trapdoor function to compute $u \leftarrow \text{PHT.}\mathsf{Invert}_{r^*, pk, a^*, x, s}(v)$. Output $\sigma = (u, x, b)$ as the signature.

$\mathsf{Ver}(vk, \mu, \sigma)$ On input message $\mu \in \{0,1\}^{\alpha}$, verification key vk, signature $\sigma = (u, x, b)$:

1. Compute $\mathbf{y} \leftarrow \text{CH.}\mathsf{Hash}(x, \mu)$.
2. Compute $a^* \leftarrow \mathsf{Eval}_{pk}^{func}(g, \mathbf{a}, (\mathbf{y}, b))$
3. Verify that the $\text{PHT.}\mathsf{Prop}(x)$
   $\text{PHT.}\mathsf{Prop}(u) \leq s$, and that $\text{PHT.}f_{pk, a^*, x}(u) = v$.

Correctness. Since $b = \mathsf{W\text{–}PRF}(s, \mathbf{y})$, we have that $g(s, (\mathbf{y}, b)) = 1 - b - b = 1 - 2b = \pm 1$ whenever $b \in \{0,1\}$. Since this will always be an invertible element of the ring $\mathcal{T} = R_q$, we have that $\text{PHT.}\mathsf{Invert}$ will invert successfully, and so verification will succeed with all but negligible probability.

3.3 Security

Security. We now prove that the signature scheme in Section 3.2 is secure.

Theorem 3.1. *Let $\mathcal{A}$ be a PPT adversary which breaks eu-acma security of the scheme in Section 3.2 in time t with advantage ϵ. Then there exists an $\mathcal{A}'$ which runs in time $t + \text{poly}(\lambda)$ and breaks the security of one of the weak PRF, the chameleon hash function and the PHTDF with advantage at least $\epsilon/3$.*

Proof. We classify a successfully forging adversary into one of three mutually exclusive categories, and show that each category of adversary can be used for a successful attack on one of the cryptographic primitives underlying our scheme.

Our reduction then proceeds by guessing uniformly at random which of these three categories our adversary belongs to and running the attack below corresponding to that category of adversary. It is easy to see that we will succeed in breaking one of the underlying cryptographic primitives with advantage at least $\epsilon/3 - \text{negl}(\lambda)$.

To complete our proof, we proceed to describe these categories of adversaries and how we use them to mount an attack on the underlying cryptographic primitives that succeeds except with probability negligible in the underlying security parameter λ.

Weak PRF. First, we consider the case that, conditioned on having forged successfully, the adversary outputs μ, $\sigma = (u, x, b)$ such that $\mathsf{W\text{–}PRF}(\mathbf{z}, (\mathsf{Hash}(x, \mu)) = b$. For this case, we use $\mathcal{A}$ to help us succeed in the underlying $\mathsf{W\text{–}PRF}$ security game against some challenger.

We change our behavior in the signature scheme's security game as follows:

1. Instead of encoding the challenger's actual $\mathsf{W\text{–}PRF}$ secret key (which we do not know), we set the secret key $\mathbf{z} = (\mathbf{s}, \mathbf{z}_{k-1}, \mathbf{z}_k)$ in our scheme to a dummy secret key of $(\mathbf{0}, 0, 0)$ in Gen, which will always allow us to sign regardless of the value of the $b = \mathsf{W\text{–}PRF}(\mathbf{s}, \mathbf{y})$.
   Since the a_i output by $\mathsf{PHT.GenTrap}$ are statistically close to uniform, independent of the $\mathbf{z}$ encoded in them, and everything else in the public key is completely independent of $\mathbf{z}$, this change is statistically indistinguishable.
2. Given a query for message μ, we send a query to the $\mathsf{W\text{–}PRF}$ challenger, receiving back a uniform random $\mathbf{y} \in \mathcal{T}^{w-1}$, $b = \mathsf{W\text{–}PRF}(\mathbf{s}, \mathbf{y})$ for some unknown $\mathbf{s}$. We then compute $x \leftarrow \mathsf{Hash}^{-1}(\mu, \mathbf{y})$ as usual. Since $\mathbf{y}$ is uniform, this is distributed within negligible statistical distance of the manner it is chosen actual signature scheme. Moreover, since by our condition on $\mathsf{W\text{–}PRF}$ we will have that $g(\mathbf{z}, (\mathsf{Hash}(x, \mu), b)) = 1 - z_{\kappa - b} - 0 = 1$, we can invert successfully and sign as usual.

As a result, our behavior in this game is statistically indistinguishable from that in the real game. At the end, if the adversary fails to output a valid signature, we choose a message μ^* and bit b^* uniformly at random and send them to the $\mathsf{W\text{–}PRF}$ challenger as our guess. Otherwise, if the adversary is successful and outputs a valid signature $\sigma = (u^*, x^*, b^*)$, we send that μ^*, b^* to the challenger. Since the adversary will correctly predict b, we succeed in guessing the output of the $\mathsf{W\text{–}PRF}$ at μ^*.

Otherwise, a successfully forging $\mathcal{A}$ outputs μ^*, $\sigma^* = (u^*, x^*, b^*)$ such that $\mathsf{W\text{–}PRF}(\mathbf{z}, (\mathsf{Hash}(x^*, \mu^*)) \neq b^*$. In this case, we consider whether or not the forgery was achieved based on a hash collision.

Chameleon Hash Collision. We consider the case that $\mathcal{A}$ finds a hash collision by successfully outputting $\mu^*, \sigma = (u^*, x^*, b^*)$ such that $\mathsf{Hash}(x^*, \mu^*) = \mathsf{Hash}(x', \mu')$ for some x', μ' it already queried on, conditioned on a successful forgery and an unsuccessful weak PRF prediction. We now show how to use $\mathcal{A}$ to break the collision-resistance of the chameleon hash function family. We change our behavior in the security game as follows:

1. In Gen, we receive the evaluation key ek for the chameleon hash function from a challenger instead of generating it (along with a trapdoor) ourselves. The public key itself remains exactly the same (although we no longer have td in our secret key), so the adversary's view does not change.
2. In Step 1 of Sign, we instead sample $x \leftarrow \mathcal{X}$, and then compute $\mathbf{y} = \mathsf{Hash}(x, \mu)$. By the uniformity property of the collision-resistance hash function, this is within statistically negligible distance of the manner x and $\mathbf{y}$ are evaluated in the actual scheme.

Since our behavior is statistically indistinguishable from that in the real game, $\mathcal{A}$ will still output a forgery such that $\mathsf{Hash}(x^*, \mu^*) = \mathsf{Hash}(x', \mu')$ for some previously output x', μ'. We can then send x', x^*, μ', μ^*, succeeding in breaking the collision-resistance of the chameleon hash function.

PHTDF Collision When Punctured. Finally, if $\mathcal{A}$ has forged successfully, but has neither successfully predicting the weak PRF output nor output a successfull chameleon hash collision, we can break collision-resistance when punctured (CRP)-security of the PHTDF. We change our behavior in the security game as follows:

1. During Gen, we choose a message μ' uniformly at random. Then, instead of choosing $v \leftarrow \mathcal{V}$, we invert the process, sampling $u' \leftarrow D_{\mathcal{U}}$, $x' \leftarrow \mathcal{X}$. We then let $\mathbf{y}' = \mathrm{CH}.\mathsf{Hash}(x', \mu')$, $b = \mathsf{W\text{–}PRF}(\mathbf{s}, \mathbf{y}')$, and compute $a' \leftarrow \mathsf{Eval}_{pk}^{func}(g, \mathbf{a}, (\mathbf{y}', 1-b))$.
   Finally, we set $v \leftarrow f_{pk,a',x'}(u')$, and hold onto u' to use in forming our collision. By the statisticial distributional equivalence of inversion property of the PHTDF, this is statistically indistinguishable from having sampled $v \leftarrow \mathcal{V}$ as in the real security game. Moreover, we have that $g((\mathbf{s}, z^{(0)}, z^{(1)}), (\mathbf{y}', (1-b))) = 0$.

If the adversary outputs a successful forgery $(\mu^*, \sigma^* = (x^*, u^*, b^*))$, where $\mathsf{W\text{–}PRF}(\mathbf{s}, (\mathsf{Hash}(x^*, \mu^*)) = 1 - b$, then because $\mathcal{A}$ has no knowledge of the x' we sampled above, with all but probability $1/|\mathcal{X}| = \mathrm{negl}(\lambda)$, $x^* \neq x'$. If they are in fact not equal, we can output $u', x', \mathbf{y}', u^*, x^*, \mathbf{y}^* = \mathsf{Hash}(\mu^*, x^*)$ as our collision, breaking the CRP security of the underlying PHTDF except with probability negligible in λ.

3.4 Efficient Evaluation of g

Here we discuss how to homomorphically evaluate the function

$$g(\mathbf{z}, (\mathbf{y}, b)) = (1 - z_{2-b}) - \mathsf{W\text{–}PRF}(\mathbf{z}, \mathbf{y})$$

using a cyclotomic ring-based instantiation of PHTDFs in a manner that is (somewhat) efficient in terms of both time and noise growth, for the specific case that $\mathsf{W\text{–}PRF}$ is the LWR function. This essentially boils down to efficiently evaluating the LWR function, as once $b = lwr_{k,Q,2}$ has been homomorphically computed, the remaining operations consist solely of additions. Our method of evaluation is very close to that of Ducas and Micciancio [DM15], but avoids any use of the general lattice setting for efficiency.

Lagrange Interpolation over R_q To evaluate the rounding part of the $\mathsf{LWR}_{n,Q,2}$ function homomorphically, we will need to be able to evaluate a function over R_q (where $R = \mathcal{O}_m$) that sends ζ^i to 0 when $i \in [-Q/4, Q/4)$, and sends ζ^i to 1 otherwise.

To homomorphically evaluate $\mathsf{LWR}_{n,Q,2}$ (where $Q = 2^\ell = O(\lambda)$ for $\ell = \lceil \log_2(\lambda) \rceil$), it will be easiest to work directly over the 2^ℓth cyclotomic ring $R = \mathcal{O}[2^\ell]$; this restricts us to odd moduli q in the actual PHTDF instantiation over R_q. The actual evaluation process will then be as described in the work of Ducas and Micciancio [DM15], using $q = 3^\alpha$ for some α. We restate the relevant result here for LWR, recall again that evaluating LWR homomorphically over a PHTDF instantiated in the ring setting. is morally equivalent to evaluating the decryption function. Note that the conditions on parameter s assumes the use of the sampling ideas found Section 4 to minimize the size of parameter s, for modulus $q = 3^\alpha$.

Theorem 3.2 ([DM15]). *Let PHT be instantiated over R_q, where $R = \mathcal{O}[m]$, where q and m are coprime. Then $\mathsf{LWR}_{n,Q,2}$ is admissible with parameter $s = O(n^{1.5} \log Q)$, and can be evaluated with $O(n \log Q)$ homomorphic operations, with the secret key $\mathbf{z} \in \mathbb{Z}_Q^n$ encoded in $\kappa - 2 = n \log Q$ functions $a_1, \ldots, a_{\kappa-2}$ with associated tags $\tilde{z}_1, \ldots, \tilde{z}_{\kappa-2}$.*

We finally have the following corollary for a concrete instantiation using the PHTDF of Alperin-Sheriff (see Section 2.5).

Corollary 3.3 ([DM15]). *Let PHT be instantiated over R_q, where $R = \mathcal{O}[m]$, where q and m are coprime. Then the above signature scheme is secure as long as both of the following hold:*

- $\mathsf{LWR}_{n,Q,2}$ *is secure for $n, Q = O(\lambda)$*
- RSIS *is secure over m with parameter $\beta = \tilde{O}(n^{7/2})$.*

4 Reducing Trapdoor Growth

Here we give a sampling algorithm that is nearly as efficient as bit decomposition, has (slightly) *lower* rates of growth than samples generated as discrete Gaussians, and which requires exactly 2 random coins per element of the vector being sampled.

We later show how to reduce the total number of random coins needed per signature/encryption etc. to $O(\lambda)$.

4.1 Distribution Definition and Properties

Definition 4.1. *Let $q \geq 2 \in \mathbb{Z}$. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ such that linear combination of its columns generate all of $\mathbb{Z}_q^n$, and let $\mathbf{u} \in \mathbb{Z}_q^n$. We define $B_{\Lambda_{\mathbf{u}}^\perp(\mathbf{A}),s}$ as the distribution which outputs $\mathbf{z} \in \mathbb{Z}_q^m$, where each element of $\mathbf{z}$ is chosen to be some integer $x \in [-s, \ldots, +s]$ with probability $\frac{(s+1-|x|)}{(s+1)^2}$, conditioned on $\mathbf{Az} = \mathbf{u}$.*

For the remainder of Section 4, we focus only on $B_{\Lambda_u^\perp(\mathbf{g}^t),p-1}$, where $q = p^k$ and $\mathbf{g}^t$ is the "gadget" vector $\mathbf{g}^t = [1, p, \ldots, p^{k-1}] \in \mathbb{Z}_q^{1 \times k}$.

Lemma 4.2. *For all $u \in \mathbb{Z}_q$, $B_{\Lambda_u^\perp(\mathbf{g}^t),p-1}$ is a centered subgaussian with parameter $(p-1)\sqrt{k}$.*

Proof. Denote by $\mathbf{g}^{(j)}$ the first j vectors of $\mathbf{g}$. To show that the output $\mathbf{z} \leftarrow B_{\Lambda_u^\perp((\mathbf{g}^{(j)})^t),1}$ is centered, we proceed by induction on the first j vectors of $\mathbf{z}$, with the inductive assumption being that over the integers, $\langle \mathbf{g}^{(j)}, \mathbf{z} \rangle = u$ with probability $\max(0, \frac{p^j - |u|}{p^{2j}})$. It is not difficult to see that this inductive hypothesis immediately implies centering.

For $j = 1$, the condition is satisfied by the definition of the distribution. Now, we assume our hypothesis is true for all $k < j$. Let $\mathbf{x} = (\tilde{\mathbf{x}} \in \mathbb{Z}_q^{k-1}, x_k)$, and let Let $y = \langle \mathbf{g}^{(k)}, \mathbf{x} \rangle$ (over the integers), and let $y^* = \langle \mathbf{g}^{(k-1)}, \tilde{\mathbf{x}} \rangle$, so $y = y^* + x_k$. Let $u = y \bmod p^{k-1}$ be the unique coset representative of y in $\mathbb{Z}_{p^{k-1}}$, and let t be such that $p^{k-1}t + u = y$.

Then either $y^* = u$, $x^k = t$, or $y^* = u - \mathsf{sign}(u)p^{k-1}$ and $x^k = t + \mathsf{sign}(u)$. As a result, the probability over the choice of $\mathbf{x}$ that $\langle \mathbf{g}^{(k)}, \mathbf{x} \rangle = y$ is

$$\begin{aligned}\rho &= \frac{(p^{k-1} - |u|)(p - |t|) + (p^{k-1} - |u - \mathsf{sign}(u)p^{k-1}|)(p - |t + \mathsf{sign}(u)|)}{p^{2k}} \\ &= \frac{p^k - u - p^{k-1}t}{p^{2k}} = \frac{p^k - y}{p^{2k}}.\end{aligned}$$

Thus, the inductive assumption is true for $j = k$, so the distribution is indeed centered conditioned on any u.

To see that it is subgaussian with parameter $(p-1)\sqrt{k}$, it is sufficient to note that for any $\mathbf{x}$ in the support of B, $\max_{\|\mathbf{u}\|=1} \langle \mathbf{u}, \mathbf{x} \rangle \leq (p-1)\sqrt{k}$.

By the standard properties of subgaussian distributions and the form of the decomposition of $\mathbf{G}$, we immediately have the following corollary, which allows us to achieve noise growth rate $\tilde{O}((p-1)\sqrt{n})$ per multiply instead of $\tilde{O}((p-1)n)$ (assuming we can efficiently sample the distribution).

Corollary 4.3. *For all $u \in \mathbb{Z}_q$, $q = p^k$ $B_{\Lambda_{\mathbf{u}}^\perp(\mathbf{G}),(p-1)}$ is a centered subgaussian with parameter $(p-1)\sqrt{nk}$.*

The following lemma is necessary for showing how to actually sample $B_{\Lambda_u^\perp(\mathbf{G}),p-1}$ in practice.

Lemma 4.4. *For $\mathbf{x} \leftarrow B^k$, $\langle \mathbf{g}, \mathbf{x} \rangle$ is uniformly distributed modulo q.*

Proof. It is easy to see that the kth element of $\mathbf{x}$ randomizes the kth least significant mod-p digit of $\langle \mathbf{g}, \mathbf{x} \rangle + q/2$ modulo q, so $\langle \mathbf{g}, \mathbf{x} \rangle$ must be uniform modulo q.

Sampling the Distribution. The distribution can be sampled with two different approaches, or a hybrid of the two, in the same manner used for sampling a discrete Gaussian distribution over $\Lambda_{\mathbf{u}}^\perp(\mathbf{G})$ as described in [MP12]. The advantage here is that instead of having to sample any discrete Gaussians in each coordinate with

the various disadvantages thereof, we can sample $B_{\mathbb{Z},p-1}$ in a very simple manner (and sample $B_{\mathbb{Z}^k,p-1}$ by sampling each coordinate independently according to $B_{\mathbb{Z},p-1}$).

Concretely, to generate a sample $B_{\mathbb{Z},p-1}$, we receive 2 (pseudo)random elements in $[0,1,\ldots,p-1]$, viewed as a single element $r \in [0,1,\ldots,p^2-1]$, and set the output x to be k with probability $|p-k|p^2$, mapping values in $[0,1,\ldots,p^2-1]$ in the canonical manner. In particular,

$$x = \begin{cases} -k & \text{if } \frac{(p-(k)-1))(p-k)}{2} \leq r < \frac{(p-k)(p-k+1)}{2} \text{ and } r < \lceil p^2/2 \rceil \\ k & \text{if } \frac{(p-(k)-1))(p-k)}{2} \leq p^2-1-r < \frac{(p-k)(p-k+1)}{2} \text{ and } r \geq \lceil p^2/2 \rceil \end{cases}$$

Given the above algorithm for sampling $B_{\mathbb{Z}^k,p-1}$, as stated, we have two options at the extremes of storage/parallelism trade-offs, as well as various hybrids between the two. The first is to precompute, sampling and storing many independent samples $\mathbf{x} \leftarrow B_{\mathbb{Z}^k,p-1}$. Assuming that values u for which samples of $B_{\Lambda_u^\perp(\mathbf{g}^t),p-1}$ are required are uniformly distributed modulo q (which will generally be the case in our desired application), and that we will require tq samples overall, by a coupon-collecting argument, we would have to store about $tq \log q$ samples to ensure we have enough on average, and tq^2 to ensure we have enough with overwhelming probability. For the second approach, we sample each coordinate one at a time in a randomized nearest-plane manner [GPV08, MP12]. Specifically, for $i = 1,\ldots,k$: choose $x_i \leftarrow B_{p\mathbb{Z}+u,s}$, and let $u \leftarrow (u-x_i)/p \in \mathbb{Z}$. This approach requires $\log q$ steps, but this is essentially no worse than bit decomposition because sampling B is so efficient. There is also a hybrid between the two approaches that involves choosing ℓ coordinates of $\mathbf{x}$ at a time for $1 < \ell < k$. For further details, see [MP12]. The key takeaway is that we can avoid the additional time and/or space costs of rejection sampling (which is potentially problematic in applications because it does not run in so-called "constant time") or storing tables of results that are required for sampling discrete Gaussians.

4.2 How to Inject Verifiable Randomness

Above, we showed how randomizing the computation of the homomorphic trapdoor functions can significantly reduce noise growth without significantly increasing computation time.

Here we discuss a semi-generic method of injecting verifiable randomness into the homomorphic computation itself, that can be applied in essentially all cryptographic schemes using key-homomorphic trapdoor functions. We also discuss a further optimized method of randomness injection that applies to **G**-based signature schemes that use chameleon hash functions to randomize the messages being signed.

Generic Method The trivial method of injecting verifiable randomness into the homomorphic computation is to simply include all of the random bits needed as part of the ciphertext or signature. While this approach may be acceptable

for very simple functions, for more complex functions it will increase the size of the ciphertext or signature by an unacceptably large amount.

Instead, it is sufficient to simply include λ bits in each signature or ciphertext, where λ is the desired security parameter, and to stretch those bits to the necessary length with a pseudorandom number generator specified as a public parameter of the scheme in question.

Interestingly, a cryptographically secure pseudorandom number generator is *not* required for this purpose, as we do not need the generated bits to be computationally indistinguishable from random under *all* polynomial-time statistical tests. Instead, the role of the PRG here is similar to that of the PRF used by Hohenberger and Waters [HW09] in their short, stateless signature scheme, where they simply needed the output of the PRF to be randomly distributed with respect to several non-adversarial statistical tests.

Indeed, for our bounds from Section 4.1 to hold for a given pseudorandom number generator G, we simply need $B_{\Lambda_u^\perp(\mathbf{g}^t)}$ to remain a centered subgaussian with parameter $\sqrt{k}$ when the bits used in sampling B come from $G(\mathbf{x})$, where $\mathbf{x}$ is a seed chosen uniformly at random. As a result, any heuristic pseudorandom number generator suitable for Monte Carlo simulations ought to be satisfactory.

4.3 Using Chameleon Hash Functions

While the method in Section 4.2 will already reduce the overall length of the outputs, it does have the downside of having to include λ bits of randomness in each individual signature or encryption. However, we can do better in signature schemes that already use chameleon hash functions to randomize the messages being signed. In this case, we show that it is sufficient to simply include an additional λ bits of randomness in the public key. We stress that the signatures remain stateless, as we reuse the same λ bits of verifiable randomness in the homomomorphic computation of each individual signature.

We now present a (semi)-formal transformation. While the conditions required of the transformation seem very specific, they do apply to several previous lattice-based signature schemes [DM14, Alp15] as well as to our scheme.

Let $\mathrm{SIG} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a secure lattice-based signature scheme which applies a secure hash function $ch = (\mathsf{Gen}, \mathsf{Hash}, \mathsf{Hash}^{-1})$ to the messages and then deterministically evaluates a $\mathbf{G}$-based key-homomorphic trapdoor function requiring ℓ invocations of $\mathbf{G}^{-1}$ (meaning a total of $2\ell n^2 k^2$ random bits) in the signing and verification algorithms. We construct $\mathrm{SIG}' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ as follows:

$\mathsf{Gen}'(1^\lambda)$ Run Gen, sample λ random bits, choose some secure pseudorandom generator PRG and output the λ random bits $\mathbf{b}$ along with a description of the pseudorandom generator.

$\mathsf{Sign}'(\mu)$ Run Sign, but evaluate the $\mathbf{G}$-based key-homomorphic functions using the $2\ell n^2 k^2$ bits output by $\mathsf{PRG}(\mathbf{b})$. Output signature σ.

$\mathsf{Ver}'((\mu, \mathbf{SIG}))$ Run Ver, but evaluate the $\mathbf{G}$-based key-homomorphic functions using the $2\ell n^2 k^2$ bits output by $\mathsf{PRG}(\mathbf{b})$. Output the result of Ver.

For stating our result, it will be convenient to view the signature scheme as being instantiated by a PHTDF which evaluates some function g admissible with parameter s. All standard model lattice-based signature schemes that we are aware of which use $\mathbf{G}$-based trapdoors can be described in this manner.

Theorem 4.5 (Informal). *If SIG is evaluates a function admissible with parameter s, then $SIG' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ evaluates a function admissible with parameter $\sqrt{s}$.*

Proof. That the scheme remains secure is straightforward. The only change is the manner in which $\mathbf{G}^{-1}$ is evaluated, and this depends only on the extra λ bits $\mathbf{b}$ in the public key and the pseudorandom generator PRG. In particular, the evaluation is performed entirely independently of the scheme's secret key. As a result, this change leaks no additional information, and so the scheme remains secure.

It remains to show that the function is admissible with parameter $\sqrt{s}$. If $\mathbf{G}^{-1}$ were evaluated with truly random bits, this would informally follow by noting the growth rate of $\mathbf{G}^{-1}$ in Corollary 4.3 versus the growth rate from bit decomposition.

First, consider an individual message query μ by the adversary. Since the scheme uses chameleon hash functions, the actual (hashed) messages being signed are randomized, and in particular, chosen in a manner independent of the λ random bits in the public key.

As a result, we may view the message as having been chosen first, and the λ random bits as having been chosen subsequently and uniformly at random and independently. As a result, as above, the probability that the function is admissible with parameter $\sqrt{s}$ using $\mathsf{PRG}(\mathbf{b})$ as our source of randomness is at most negligibly smaller than the probability that the function is admissible with parameter $\sqrt{s}$ using truly random bits as our source of randomness. To reiterate, this is because a noticeable difference in the probabilities would mean that we have a statistical test that is entirely independent of the seed $\mathbf{b}$, and would thus break the assumed pseudorandomness of PRG.

A simple union bound over the $Q = \mathrm{poly}(\lambda)$ message queries means that the scheme remains correct except with negligible probability.

References

[ABG+14] Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. Candidate weak pseudorandom functions in $\mathrm{ac}^0 \circ \mathrm{mod}_2$. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 251–260, 2014.

[Ajt04] Miklós Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.

[AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties a nd applications. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 57–74, 2013.

[Alp15] Jacob Alperin-Sheriff. Short signatures with short public keys from homomorphic trapdoor functions. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 236–255, 2015.

[AP13] Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasi-linear time. In *CRYPTO (1)*, pages 1–20, 2013.

[ASA16] Jacob Alperin-Sheriff and Daniel Apon. Dimension-preserving reductions from lwe to lwr. Cryptology ePrint Archive, Report 2016/589, 2016. `http://eprint.iacr.org/2016/589`.

[ASP14] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In *CRYPTO (1)*, pages 297–314, 2014.

[BB08] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.

[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.

[BGM+16] Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. On the hardness of learning with rounding over small modulus. In *Theory of Cryptography - 13th International Conference, TCC 2 016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 209–224, 2016.

[BHJ+14] Florian Böhl, Dennis Hofheinz, Tibor Jager, Jessica Koch, and Christoph Striecks. Confined guessing: New signatures from standard assumptions. *Journal of Cryptology*, pages 1–33, 2014.

[BKKP15] Olivier Blazy, Saqib A. Kakvi, Eike Kiltz, and Jiaxin Pan. Tightly-secure signatures from chameleon hash functions. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 256–279, 2015.

[BL16] Xavier Boyen and Qinyi Li. Towards tightly secure lattice short signature and id-based encryption. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, pages 404–434, 2016.

[BLL+15] Shi Bai, Adeline Langlois, Tancrède Lepoint, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: Using the rényi divergence rather than the statistical distance. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, pages 3–24, 2015.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *Public Key Cryptography*, pages 499–517, 2010.

[BP14] Abhishek Banerjee and Chris Peikert. New and improved key-homomorphic pseudorandom functions. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 353–370, 2014.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, pages 719–737, 2012.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 1–12, 2014.

[CDW16] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short stickelberger class relations and application to ideal-svp. *IACR Cryptology ePrint Archive*, 2016:885, 2016.

[CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. Cryptology ePrint Archive, Report 2016/870, 2016.

[CHKP12] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. *J. Cryptology*, 25(4):601–639, 2012. Preliminary version in Eurocrypt 2010.

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *ASIACRYPT*, pages 1–20, 2011.

[CW13] Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure IBE and dual system groups. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 435–460, 2013.

[DG14] Nagarjun C. Dwarakanath and Steven D. Galbraith. Sampling from discrete gaussians for lattice-based cryptography on a constrained device. *Appl. Algebra Eng. Commun. Comput.*, 25(3):159–180, 2014.

[DM14] Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 335–352, 2014.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 617–640, 2015.

[DN02] Ivan Damgård and Jesper Buus Nielsen. Expanding pseudorandom functions; or: From known-plaintext security to chosen-plaintext security. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, pages 449–464, 2002.

[DS15] Nico Döttling and Dominique Schröder. Efficient pseudorandom functions via on-the-fly adaptation. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 329–350, 2015.

[Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `http://crypto.stanford.edu/craig`.

[GHR99] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In *EUROCRYPT*, pages 123–139, 1999.

[GHS12a] Craig Gentry, Shai Halevi, and Nigel P. Smart. Better bootstrapping in fully homomorphic encryption. In *Public Key Cryptography*, pages 1–16, 2012.

[GHS12b] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In *CRYPTO*, pages 850–867, 2012.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[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.

[GV15] Sergey Gorbunov and Dhinakaran Vinayagamurthy. Riding on asymmetry: Efficient ABE for branching programs. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, pages 550–574, 2015.

[GVW15a] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 503–523, 2015.

[GVW15b] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 469–477, 2015.

[HAO15] Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. Packing messages and optimizing bootstrapping in GSW-FHE. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 699–715, 2015.

[HS15] Shai Halevi and Victor Shoup. Bootstrapping for helib. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 641–670, 2015.

[HW09] Susan Hohenberger and Brent Waters. Short and stateless signatures from the RSA assumption. In *CRYPTO*, pages 654–670, 2009.

[KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS*, 2000.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS 2003, Washington, DC, USA, October 27-30, 2003*, pages 155–164, 2003.

[LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptography*, 75(3):565–599, 2015.

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

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. Preliminary version in FOCS 2004.

[NR99] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999. Preliminary version in FOCS 1995.

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. Preliminary version in FOCS 1997.

[NRR02] Moni Naor, Omer Reingold, and Alon Rosen. Pseudorandom functions and factoring. *SIAM J. Comput.*, 31(5):1383–1404, 2002. Preliminary version in STOC 2000.

[OvdPS15] Emmanuela Orsini, Joop van de Pol, and Nigel P. Smart. Bootstrapping BGV ciphertexts with a wider choice of p and q. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 673–698, 2015.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):1–40, 2009. Preliminary version in STOC 2005.

[RR94] Alexander A. Razborov and Steven Rudich. Natural proofs. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 204–213, 1994.

[RT92] John H. Reif and Stephen R. Tate. On threshold circuits and polynomial computation. *SIAM J. Comput.*, 21(5):896–908, 1992.

[SE94] Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathmatical Programming*, 66:181–199, 1994.

[ST01] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In *CRYPTO*, pages 355–367, 2001.

[Ver12] Roman Vershynin. *Compressed Sensing, Theory and Applications*, chapter 5, pages 210–268. Cambridge University Press, 2012. Available at `http://www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf`.