Appendixes

1 Appendix A

Before proving the correctness and estimating the number of rejections that the algorithm needs, we recall the notion of the smoothing parameter of lattices, which was introduced by Micciancio and Regev in [7]. The simplest lattice is the one-dimensional integer lattice $\mathbb{Z}$. In fact, we only need the notion of the smoothing parameter of lattice $\mathbb{Z}$.

Definition 1 (Smoothing Parameter [7]). *Let $\epsilon > 0$ be a positive real. The smoothing parameter of lattice $\mathbb{Z}$, denoted by $\eta_\epsilon(\mathbb{Z})$, is defined to be the smallest real s such that $\rho_{1/s}(\mathbb{Z} \setminus \{0\}) \leq \epsilon$.*

Lemma 1 (Gaussian Measure on Translates of the Lattice [7]). *For any (small) real $\epsilon > 0$ and $c \in \mathbb{R}$, if $\sigma \geq \eta_\epsilon(\mathbb{Z})$, then $\rho_{\sigma,c}(\mathbb{Z}) = [(1-\epsilon)/(1+\epsilon), 1] \cdot \rho_\sigma(\mathbb{Z})$.*

Lemma 2. *If $q \geq 1$ and $0 < c < 1$, for any integer z, then there exists a unique tuple (x, y, s) with $x \in \mathbb{Z}^+$, $y \in [0, \lceil q \rceil)$ and $s \in \{-1, 1\}$, such that $z = s(\lceil xq + sc \rceil + y)$ and $\lceil xq + sc \rceil - xq - sc + y < q$.*

Proof. For any integer z, there exist $x \geq 0$ and $y \in [0, \lceil q \rceil)$ such that $z = s(\lceil xq + sc \rceil + y)$, where $s = 1$ if $z > 0$ and $s = -1$ if $z \leq 0$. It is clear that $\lceil xq+sc \rceil - xq - sc + y < q$ if $y < \lceil q \rceil - 1$. Otherwise, if $y = \lceil q \rceil - 1$ and $\lceil xq + sc \rceil - xq - sc + y \geq q$, then $\lceil xq + sc \rceil + \lceil q \rceil - 1 \geq q + xq + sc$. Therefore, by taking $x' = x+1$ and $y' = 0$ we have $\lceil x'q+sc \rceil + y' = \lceil xq+q+sc \rceil = \lceil xq+sc \rceil + \lceil q \rceil - 1 = \lceil xq+sc \rceil + y$ and $\lceil x'q + sc \rceil - x'q - sc + y' = \lceil x'q + sc \rceil - x'q - sc < q$. This guarantees that there always exists a tuple (x, y, s) we desire. Assume that $\lceil x_1 q + sc \rceil + y_1 = \lceil x_2 q + sc \rceil + y_2$, where $0 \leq x_1 < x_2$ and $y_1, y_2 \in [0, \lceil q \rceil)$. Then $\lceil x_2 q + sc \rceil - \lceil x_1 q + sc \rceil = y_1 - y_2$. Since $q \geq 1$ and $0 < c < 1$, it follows that that $y_1 - y_2 = \lceil x_2 q + sc \rceil - \lceil x_1 q + sc \rceil \geq \lceil x_1 q + q + sc \rceil - \lceil x_1 q + sc \rceil \geq \lceil q \rceil - 1$. This implies that $x_2 = x_1 + 1$, $y_1 = \lceil q \rceil - 1$ and $y_2 = 0$ if there exist $0 \leq x_1 < x_2$ and $y_1, y_2 \in [0, \lceil q \rceil)$ such that $\lceil x_1 q + sc \rceil + y_1 = \lceil x_2 q + sc \rceil + y_2$. In this case, we have $\lceil x_1 q + sc \rceil + \lceil q \rceil - 1 = \lceil x_1 q + q + sc \rceil$. Hence, $\lceil x_1 q + sc \rceil - x_1 q - sc + y_1 = \lceil x_1 q + sc \rceil + \lceil q \rceil - 1 - x_1 q - sc = \lceil x_1 q + q + sc \rceil - x_1 q - sc \geq q$ and $\lceil x_2 q + sc \rceil - x_2 q - sc + y_2 = \lceil x_2 q + sc \rceil - x_2 q - sc < q$. This proves the uniqueness. □

From the proof of Lemma 2, for a given tuple (x, y, s), we see that $\lceil xq + sc \rceil - xq - sc + y \geq q$ only if $y = \lceil q \rceil - 1$. Then we have the following corollary.

Corollary 1. *Let z be a non-zero integer. Let (x, y, s) be a tuple with $x \in \mathbb{Z}^+$, $y \in [0, \lceil q \rceil)$ and $s \in \{-1, 1\}$ such that $z = s(\lceil xq + sc \rceil + y)$. If $q \geq 1$ and $0 \leq c < 1$, the probability that $\lceil xq + sc \rceil - xq - sc + y \geq q$ is at most $1/\lceil q \rceil$.*

Theorem 1. *The algorithm outputs an integer z according to discrete Gaussian distribution $D_{\mathbb{Z},\sigma,c}$ with $\sigma > \sigma_2$ and $c \in (0, 1/2]$. The expected number of rejections that the algorithm needs has an upper-bound given by $(1/\tau) \cdot \big(\lceil q \rceil \rho_{\sigma_2}(\mathbb{Z}^+)\big) / \big(q\sigma_2\sqrt{\pi/2} - 1\big)$ with $\tau = (1 - 1/\lceil q \rceil)$. In particular, if $\sigma \geq \eta_\epsilon(\mathbb{Z})$ then the expected number of of rejections has an upper-bound given by $1.47 \cdot \big(\lceil q \rceil^2 / q(\lceil q \rceil - 1)\big)$.*

Proof. Since $q = \sigma/\sigma_2 > 1$ and $0 < c \leq 1/2$, by Lemma 2, any possible output $z \in \mathbb{Z}$ can be uniquely written as $z = s(\lceil xq+sc\rceil+y)$ with $x \geq 0$ and $y \in [0, \lceil q\rceil)$, such that $\lceil xq+sc\rceil - xq - sc + y < q$. For $z \in \mathbb{Z}$, the proposal and the target distribution density function are

$$g(z) = \frac{\rho_{\sigma_2}(x)}{2\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)} \quad \text{and} \quad f(z) = f(s(\lceil xq + sc\rceil + y)) = \frac{\rho_{q\sigma_2,c}(s(\lceil xq + sc\rceil + y))}{\rho_{q\sigma_2,c}(\mathbb{Z})}$$

respectively. Note that

$$\begin{aligned}
\rho_{\sigma_2}(x) \exp\left(-\frac{2xq(y+\delta) + (y+\delta)^2}{2q^2\sigma_2^2}\right) &= \exp\left(-\frac{x^2q^2 + 2xq(y+\delta) + (y+\delta)^2}{2q^2\sigma_2^2}\right) \\
&= \exp\left(-\frac{(xq + y + \delta)^2}{2q^2\sigma_2^2}\right) \\
&= \exp\left(-\frac{(y + \lceil xq + sc\rceil - sc)^2}{2q^2\sigma_2^2}\right) = \rho_{q\sigma_2,c}(s(\lceil xq + sc\rceil + y)).
\end{aligned}$$

Furthermore, $\exp\big(-(2xq(y+\delta) + (y+\delta)^2)/2q^2\sigma_2^2\big)$ is not more than one as x and $y+\delta$ are both non-negative. For any $z = s(\lceil xq + sc\rceil + y))$, therefore, the expected number of rejections that the algorithm needs is equal to $\max\{f(z)/g(z)\} \cdot \max\{1/\tau(x)\}$, where $\tau(x)$ is the probability that $y + \delta < q$. Then, we have

$$\max \frac{f(z)}{g(z)} \leq \frac{\rho_{q\sigma_2,c}(s(\lceil xq + sc\rceil + y))}{\rho_{\sigma_2}(x)} \cdot \frac{2\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)}{\rho_{q\sigma_2,c}(\mathbb{Z})} \leq \frac{2\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)}{\rho_{q\sigma_2,c}(\mathbb{Z})} < \frac{\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)}{\rho_{q\sigma_2}(\mathbb{Z}^+\backslash\{0\})} \leq \frac{\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)}{q\sigma_2\sqrt{\pi/2} - 1}.$$

The third inequality follows from the fact that

$$\rho_{q\sigma_2,c}(\mathbb{Z}) = \rho_{q\sigma_2,c}(\mathbb{Z}^+\backslash\{0\}) + \rho_{q\sigma_2,c}(-\mathbb{Z}^+) \geq 2\rho_{q\sigma_2,c}(\mathbb{Z}^+\backslash\{0\}) > 2\rho_{q\sigma_2}(\mathbb{Z}^+\backslash\{0\})$$

for $c \in (0, \frac{1}{2}]$, where the equality holds when $c = \frac{1}{2}$. The last inequality follows from $\rho_{q\sigma_2}(\mathbb{Z}^+\backslash\{0\}) = \rho_{q\sigma_2}(\mathbb{Z}^+) - 1$ and the sum-integral comparison

$$\rho_{q\sigma_2}(\mathbb{Z}^+) \geq \int_0^{+\infty} \rho_{q\sigma_2}(x)dx = q\sigma_2\sqrt{\pi/2}.$$

In addition, by Lemma 1, if $q\sigma_2 \geq \eta_\epsilon(\mathbb{Z})$, then $\rho_{q\sigma_2,c}(\mathbb{Z}) \approx \rho_{q\sigma_2}(\mathbb{Z})$. Then we have

$$\frac{2q\rho_{\sigma_2}(\mathbb{Z}^+)}{\rho_{q\sigma_2,c}(\mathbb{Z})} \approx \frac{2q\rho_{\sigma_2}(\mathbb{Z}^+)}{\rho_{q\sigma_2}(\mathbb{Z})} \leq \frac{2q\rho_{\sigma_2}(\mathbb{Z}^+)}{2q\sigma_2\sqrt{\pi/2}} \leq \frac{\rho_{\sigma_2}(\mathbb{Z}^+)}{\sigma_2\sqrt{\pi/2}} \approx 1.47,$$

where the first inequality follows from the sum-integral comparison

$$\rho_{q\sigma_2}(\mathbb{Z}) \geq \int_{-\infty}^{+\infty} \rho_{q\sigma_2}(x)dx = 2k\sigma_2\sqrt{\pi/2}.$$

Finally, for $x \geq 0$, by Corollary 1, we have $\tau(x) \geq 1 - 1/\lceil q\rceil$. Let $\tau = \min \tau(x) = 1 - 1/\lceil q\rceil$. The expected number of rejections has an upper-bound given by $(1/\tau) \cdot \big(\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)\big)/\big(q\sigma_2\sqrt{\pi/2} - 1\big)$. In particular, by Lemma 1, if $\sigma = q\sigma_2 \geq \eta_\epsilon(\mathbb{Z})$ then $\rho_{q\sigma_2,c}(\mathbb{Z}) \approx \rho_{q\sigma_2}(\mathbb{Z})$. Thus, we have $2\lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)/\rho_{q\sigma_2,c}(\mathbb{Z}) \leq \lceil q\rceil \rho_{\sigma_2}(\mathbb{Z}^+)/q\sigma_2\sqrt{\pi/2}$ and the expected number of rejections has an upper-bound given by $\big(\lceil q\rceil^2/q(\lceil q\rceil - 1)\big) \cdot \big(\rho_{\sigma_2}(\mathbb{Z}^+)/\sigma_2\sqrt{\pi/2}\big) \approx 1.47 \cdot \big(\lceil q\rceil^2/q(\lceil q\rceil - 1)\big)$. □

2 Appendix B

Time independent is not constant-time. The former means more than the latter. As mentioned in section 1, ensuring constant execution time is one of ways in which we achieve the time independence. Since rejection sampling is inherently costly to turn into constant-time algorithms due to the fact that they are probabilistically rejecting samples, our proposed algorithms cannot be constant-time. In this section, we show that our algorithm can be adapted for meeting the requirement of time independence, and thus it has resistance against timing attacks.

The Sample$\mathbb{Z}$ algorithm proposed by Ducas et al. in [2], which is called Bernoulli sampling by convention, is susceptible to timing attacks because of the implementation of the binary discrete Gaussian distribution $D_{\mathbb{Z}^+,\sigma_2}$ and the process of computing the exponential function $\exp\{-(y^2 + 2kxy)/2k^2\sigma_2^2\}$ trough a pre-computed look-up table [1].

We assume that double precision floating-point arithmetic does not leak the values of the operands. Although a subnormal (denormal) floating point value and a value with a zero exponent exhibit different timing behaviors compared with normal floating point values according to recent observations [6], it is not hard for us to avoid using these special values in our algorithms and there is no more evidence that the values of operands of double precision floating-point arithmetic can be recovered from those timing behaviors. Our algorithm is an 'off-centralized' version of the Bernoulli sampling, but it computes the exponential function $\exp\left(-\frac{2xq(y+\delta)+(y+\delta)^2}{2q^2\sigma_2^2}\right)$ by directly using (double precision) floating-point arithmetic, and thus it prevents information leakage caused by using the pre-computed look-up table. However, it seems to be difficult for us to give a time independent implementation of discrete Gaussian distribution $D_{\mathbb{Z}^+,\sigma_2}$. Therefore, we need to mitigate the information leakage due to implementation of $D_{\mathbb{Z}^+,\sigma_2}$.

Note that the samples from $D_{\mathbb{Z}^+,\sigma_2}$ (the values of x) can be generated in the offline phase in Algorithm 3, as the sampler for $D_{\mathbb{Z}^+,\sigma_2}$ is fixed and does not depend on the input of Algorithm 3. In this case, for a fixed sampler, Micciancio et al. suggested generating the samples in large batches in the offline phase [8]. In our algorithm, we can use this strategy simply by adding a minor performance penalty. Then one can look at the time required to generate a batch of samples from $D_{\mathbb{Z}^+,\sigma_2}$. Their aggregate generation time is sharply concentrated around the expectation, and can be made constant except with negligible probability.

The limitation of mitigation strategy suggested by Micciancio et al. is that it requires a cache of a large size to store a large batch of samples. We show that randomly shuffling small batches of samples from $D_{\mathbb{Z}^+,\sigma_2}$ also works for mitigating the information leakage. We assume that the adversary is able to exactly recover samples from $D_{\mathbb{Z}^+,\sigma_2}$ by timing (even other side-channel) attacks on the sampler for $D_{\mathbb{Z}^+,\sigma_2}$. Let d be the size of a batch, i.e., every d samples from $D_{\mathbb{Z}^+,\sigma_2}$ are randomly shuffled. On one hand, if the adversary has recovered $l\,(1 \le l \le d)$ consecutive samples from $D_{\mathbb{Z}^+,\sigma_2}$, after the random shuffling she/he can rearrange these values in the right order with probability $1/\prod_{i=1}^{l}(d-i+1)$, and then obtains l exact values of x with this probability. On the other hand, one can directly guess the values of x according to discrete Gaussian distribution $D_{\mathbb{Z},\sigma,c}$. More specifically, the probability that $x = i\,(i = 0, 1, 2, \ldots)$, denoted by $\Pr[x = i]$, i.e., the output integer $z \in [-(i+1)\lceil q\rceil - 1, -i\lceil q\rceil] \cup [i\lceil q\rceil, (i+1)\lceil q\rceil - 1]$, is equal to

$$\sum_{z=-(i+1)\lceil q\rceil-1}^{-i\lceil q\rceil} \rho_{\sigma,c}(z)/\rho_{\sigma,c}(\mathbb{Z}) + \sum_{z=i\lceil q\rceil}^{(i+1)\lceil q\rceil-1} \rho_{\sigma,c}(z)/\rho_{\sigma,c}(\mathbb{Z})$$

For a large $\lceil q\rceil$ (≥ 215) and a random $c \in [0, 1)$, one can verify that $\Pr[x = 0] \approx 0.76$, $\Pr[x = 1] \approx 0.22$, $\Pr[x = 2] \approx 0.018$ and $\Pr[x \ge 3] < 0.002$. Then the guessing strategy of the highest success rate is to guess $x = 0$ all the time. Thus, when $l \ge 2$ we say that the adversary's attacks

are invalid if

$$(\Pr[x=0])^l > 1/\prod_{i=1}^{l}(d-i+1),$$

which means that the effect of launching attacks is worse than that of guessing $x = 0$ directly. When $l = 1$, we assume that the adversary only considers determining a single value of x even if she/he has recovered a number of consecutive samples from $D_{\mathbb{Z}^+,\sigma_2}$. In this case, we say her/his attacks are invalid if $\Pr[x = i] > 1/d$ for $i \geq 0$. Furthermore, due to the probability density function of $D_{\mathbb{Z}^+,\sigma_2}$ the adversary needs $1/\sum_{i\geq 3}\rho_{\sigma_2}(i)/\rho_{\sigma_2}(\mathbb{Z}^+)$ attacks on average until getting a sample not less 3 from $D_{\mathbb{Z}^+,\sigma_2}$, and needs about $d/\sum_{i\geq 3}\rho_{\sigma_2}(i)/\rho_{\sigma_2}(\mathbb{Z}^+)$ attacks in all until recovering a value of x not less 3. Hence, if

$$\Pr[x \geq 3] > (1/d)\cdot\sum_{i\geq 3}\rho_{\sigma_2}(i)/\rho_{\sigma_2}(\mathbb{Z}^+),$$

which implies that the number of launching attacks is more than that of guessing $x \geq 3$ directly, we can also say that the adversary's attacks are invalid.

In practice, we choose $d = 56$ and use Fisher-Yates algorithm to randomly shuffle every 56 samples from $D_{\mathbb{Z}^+,\sigma_2}$. It can be verified that $d = 56$ satisfies the two probability inequalities so that the information leakage is invalid. Applying this mitigation strategy, we obtain an adapted version of our algorithm that is time independent. We tested that the performance decreases to about 9.2×10^6 samples per second in this case. The adversary will not be able to recover the secret information even if the adversary determines the exact number of restarts, because the output of of the algorithm does not depend on this number.

3 Appendix C

Karney's sampling algorithm for discrete Gaussian distributions over the integers $\mathbb{Z}$ can be realized by using C++ library 'RandomLib', in which the source code of his algorithm is encapsulated as a .hpp file named 'DiscreteNormal.hpp'[1]. 'RandomLib' also supports the generation and some basic operations of infinite precision random numbers. Moreover, the double-precision random numbers used in our experiments are sampled by using `<random>` library in C++11. On a laptop computer (Intel i7-6820hq, 8GB RAM, Ubuntu 16.04), using the g++ compiler and enabling -Os optimization option, we implemented Gentry's Sample$\mathbb{Z}$ algorithm [4] and our proposed algorithm. The source code is based on the adaptions of 'DiscreteNormal.hpp' as well as the runtime environment provided by 'RandomLib'.

Figure 1 shows the performance of our sampling algorithm compared with Gentry's and Karney's algorithm. For a discrete Gaussian distribution $D_{\mathbb{Z},\sigma,c}$, with σ ranging roughly from 4 to 2^{20}, and c picked uniformly from $[0, 1)$, one can get about 10.5×10^6 samples per second by using our algorithm. We see that a large σ has no impact on the sampling efficiency of our algorithm. For Karney's algorithm, however, there will be a degradation in performance when $\sigma \geq 2^{18}$.

Recent results on discrete Gaussian sampling precision suggested that the significand precision of 53 bits provided by double-precision floating arithmetic, is sufficient for most of the security applications [8, 9]. Hence, our algorithm is only designed and implemented for double-precision floating parameters. One could also design an adapted algorithm for higher precision using the lazy floating-point arithmetic [3].

We also implemented the sampling algorithm proposed by Micciancio et al. [8] in our computer. For example, we took $s = 82137.0$, $c = 0.423$, $k = 12$, $s_0 = 34$ and $b = 16$, its sampling speed was

[1] 'RandomLib' is available at http://randomlib.sourceforge.net/.

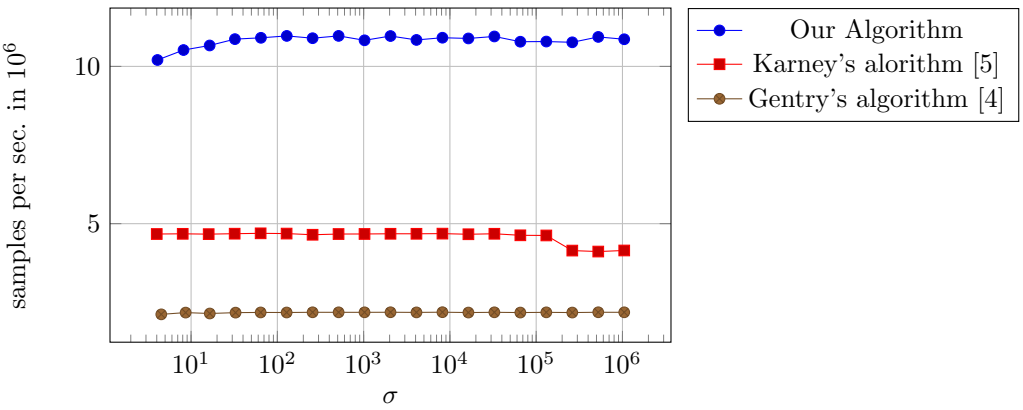

Figure 1: Performance of our sampling algorithm compared to Gentry's and Karney's algorithm

only about 1.15×10^6 samples per second at the cost of pre-computed tables of size about $2^{15.4}$ bytes. This means that our algorithm has a significant performance advantage. Although this is only a naive implementation, its results roughly agree with the experimental data presented in [8]. Micciancio et al. also suggested splitting their algorithm into online and offline phases, and they assumed the offline phase is free (by using idle times, parallel devices, FPGAs or GPUs). In this case, its performance approaches 10.0×10^6 according to their experimental data. In fact, our algorithms can also be split into online and offline phases, and we can also get a performance boost if the offline phase is free.

References

[1] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. Flush, gauss, and reload - A cache attack on the BLISS lattice-based signature scheme. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *LNCS*, pages 323–345. Springer, 2016.

[2] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *LNCS*, pages 40–56. Springer, 2013.

[3] Léo Ducas and Phong Q. Nguyen. Faster gaussian lattice sampling using lazy floating-point arithmetic. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *LNCS*, pages 415–432. Springer, 2012.

[4] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Cynthia Dwork, editor, *STOC 2008, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206. ACM, 2008.

[5] Charles F. F. Karney. Sampling exactly from the normal distribution. *ACM Trans. Math. Softw.*, 42(1):3:1–3:14, 2016.

[6] David Kohlbrenner and Hovav Shacham. On the effectiveness of mitigations against floating-point timing channels. In Engin Kirda and Thomas Ristenpart, editors, *USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017.*, pages 69–81. USENIX Association, 2017.

[7] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

[8] Daniele Micciancio and Michael Walter. Gaussian sampling over the integers: Efficient, generic, constant-time. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *LNCS*, pages 455–485. Springer, 2017.

[9] Thomas Prest. Sharper bounds in lattice-based cryptography using the rényi divergence. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *LNCS*, pages 347–374. Springer, 2017.