# TESLA: Tightly-Secure Efficient Signatures from Standard Lattices

Erdem Alkim[1], Nina Bindel[2], Johannes Buchmann[2], and Özgür Dagdelen[3] *

[1] Department of Mathemathics, Ege University, Turkey
`erdemalkim@gmail.com`
[2] Technische Universität Darmstadt, Germany
`nbindel@cdc.informatik.tu-darmstadt.de, buchmann@cdc.informatik.tu-darmstadt.de`
[3] BridgingIT GmbH, Germany
`oezdagdelen@googlemail.com`

**Abstract.** Generally, lattice-based cryptographic primitives offer good performance and allow for strong security reductions. However, the most efficient current lattice-based signature schemes sacrifice (part of its) security to achieve good performance: first, security is based on *ideal lattice problems*, that might not be as hard as *standard lattice problems*. Secondly, the security reductions of the most efficient schemes are *non-tight*; hence, their choices of parameters offer security merely heuristically. Moreover, lattice-based signatures are instantiated for classical adversaries, although they are based on presumably quantum hard problems. Yet, it is not known how such schemes perform in a *post-quantum world*. We bridge this gap by proving the lattice-based signature scheme TESLA to be tightly secure based on the learning with errors problem over standard lattices in the random oracle model. As such, we improve the security of the original proposal by Bai and Galbraith (CT-RSA'14) twofold; we tighten the security reduction and we minimize the underlying security assumptions. Remarkably, by enhancing the security we can improve TESLA's performance by a factor of two. Furthermore, we are first to propose parameters providing a security of 128 bits against *both* classical and quantum adversaries for a lattice-based signature scheme. Our implementation of TESLA competes well with state-of-the-art lattice-based signatures and SPHINCS (EUROCRYPT'15), the only signature scheme instantiated with quantum-hard parameters thus far.

**Keywords.** signature scheme, lattice cryptography, tight security, efficiency, quantum security

## 1 Introduction

In 1994, Shor presented a polynomial-time quantum algorithm for the factorization of integers and the solution of the discrete-logarithm problem [62]. Therefore, it is clear that all public-key cryptography that is in wide use today will fall, once a large quantum computer can be built. Post-quantum public-key cryptography addresses this by offering alternatives that—as far as we know—will not be broken by a quantum computer in polynomial time.

Unfortunately, the current state-of-the-art is far from offering practical alternatives for schemes such as RSA, DSA, or elliptic-curve-based schemes. One reason is that many post-quantum schemes suffer from practical problems such as large key sizes, long computation times, large ciphertext expansion, or large signature sizes. Another problem is that the hardness of the underlying problems is not as well understood as the hardness of factoring or the discrete logarithm problem.

We stress, however, that it is not sufficient for a cryptosystem to be based on some well-understood computational problem; the tightness of the relation — the security reduction —

* This work has been supported by the the German Research Foundation (DFG) as part of project P1 within the CRC 1119 CROSSING, by TÜBITAK under 2214-A Doctoral Research Program Grant, and by the Netherlands Organisation for Scientific Research (NWO) through Veni 2013 project 13114. Permanent ID of this document: `4f2d342337db6ce6cc0c7b44159b4598`. Date: 2015-05-20.

between the security of the scheme and the computational problem is important as well: in the security reduction from a hard problem to a signature scheme, an algorithm is constructed which solves the hard problem making black-box use of an adversary who can forge a signature of the scheme. If the success probabilities of the reduction and the adversary are about the same given roughly the same computational times, the reduction is called *tight*. Non-tight security reductions provide insight into the asymptotic security of the construction, but say little or close to nothing about the security of instantiations of the schemes with practical parameters. In order to provide a target security level, signature schemes with a non-tight security reduction must be instantiated with larger parameters to compensate the loose reduction. The importance of tight security reductions is extensively discussed, for instance, by Bellare and Rogaway [13] or by Chatterjee et al. [25].

Strong security and good performance are not independent; schemes with (tight) reductions from problems that are somewhat better understood and accepted than a more conservative choice tend to be less efficient in size and speed, in classical as well as in post-quantum cryptography. In the realm of lattice-based signatures, a trade-off between efficiency and reliability is the choice between *standard lattices* and *ideal lattices*. Schemes based on ideal lattices have much smaller public and private keys and are generally believed to give better performance. However, the additional cyclic structure of ideal lattices might be exploited by an attacker. Although recent results by Garg et al. [36], Campbell et al. [23], and Bernstein [14] do not seem to apply directly to most of the proposed ideal-lattice-based signature schemes, they raise some concerns about the security of such.

The currently leading lattice-based signature schemes, BLISS [33] and GLP [39], base their security on computational problems over ideal lattices. Moreover, the efficient instantiations of BLISS generate their secret keys similarly to NTRU [41], which induces an additional (non-lattice) assumption. The security of GLP relies on the discrete compact knapsack (DCK) problem. While DCK is a variant of the learning with errors problem over rings (R-LWE) with an aggressive choice of parameters, there is no reduction known from worst-case lattice problems to DCK (as opposed to R-LWE where such worst-case reductions are given [32,53,54]). Additionally, the security reductions underlying BLISS and GLP are highly non-tight such that their selection of parameters is based on heuristics and hence, should be taken with care. Ideally, a lattice-based signature scheme should be based on standard lattice problems with a tight security reduction from them.

Furthermore, most proposals that are expected to be post-quantum secure, choose concrete parameter instantiations considering attacks by classical adversaries. In fact, only very few papers propose parameters that offer security against attacks by quantum computers (see Table 3 in Section 6). We discuss and compare security and performance issues of lattice-based and other post-quantum signature schemes from the literature in Section 6.

**Our contributions.** We address the above issues by presenting a lattice-based signature scheme, which we call TESLA: "**T**ightly-secure, **E**fficient signature scheme from **S**tandard **LA**ttices". We note that the novelty of TESLA does not rely on its construction. In fact, the design of TESLA is essentially the signature scheme by Bai and Galbraith [10], who base the security of their scheme on *both* the learning with errors (LWE) and the smallest integer solution (SIS) problem with a *loose* security reduction. However, our contributions introduce significant theoretical and practical improvements to TESLA (and beyond). Our contributions are summarized as follows:

– We propose TESLA-128 that offers 128 bits of security against all known attacks, *excluding* attacks by a quantum computer. Our implementation of TESLA-128 is significantly faster than the high-speed implementation of the same scheme given in [65] and even key sizes are 7-25 times smaller than [65].
– We propose TESLA-256 that offers 128 bits of security against all known attacks, *including* attacks by a quantum adversary. After the stateless hash-based signature scheme SPHINCS [15], proposed at Eurocrypt 2015, TESLA-256 is the second signature scheme (and the first lattice-based scheme) to offer this level of security against attacks by quantum adversaries.

- We give a *tight* security reduction from the LWE problem on *standard* lattices in the random-oracle model. Our novel security reduction is inspired by a method of Katz and Wang [42][4]. The previous reduction by Bai and Galbraith [10] is not tight and relies on the hardness of LWE *and* SIS.
- We present high-performance software implementations of TESLA-128 and TESLA-256 targeting Intel Haswell CPUs. This software does not leak any secret information through timing and outperforms all previous standard-lattice-based signatures at comparable security levels. Furthermore, TESLA-128 performs well to ideal-lattice-based signature schemes. It is outperformed by the BLISS software presented in [34] only by a factor of two, while having shorter signature sizes. The software performance of TESLA relies on the careful choice of parameters, which is enabled by the tight security reduction from LWE and a novel implementation technique that reduces penalties from insufficient cache throughput. This technique might be of independent interest. The software will be made available soon.
- In Appendix C we present a tight security reduction from LWE to *a variant of* TESLA-256 in the *quantum random oracle model* (QROM) [20][5]. One step in this reduction requires a chameleon hash function where TESLA-256 simply uses SHA-512. We *could* have decided to implement and use a chameleon hash function instead of SHA-512 in TESLA-256, but that would have meant to significantly sacrifice performance without gaining protection against any known attack. We believe that the need for a chameleon hash function is merely an artifact of the proof and we would be surprised if our decision to use SHA-512 could be exploited in an attack. Clearly, such an attack would be a major contribution to the community's understanding of security reductions in the quantum random oracle model.

**Organization of this paper.** Section 2 briefly gives some necessary background and establishes notation. Section 3 presents the signature scheme TESLA and our novel security reduction. Section 4 analyzes the best known attacks against LWE and derives the parameters for TESLA-128 and TESLA-256. Section 5 gives some details of our software implementation. Finally, Section 6 presents performance results and concludes with a comparison between TESLA and results from the literature.

## 2 Preliminaries

### 2.1 Notation

Throughout this paper $q$ is an integer (if not stated otherwise) and the elements in the ring $\mathbb{Z}_q$ are represented by the set of integers $(-\lfloor q/2 \rfloor, \lceil q/2 \rceil]$. We denote a column vector $\mathbf{v}$ by bold lower case letters and a matrix $\mathbf{M}$ by bold upper case letters. The transpose of a vector or a matrix is denoted by $\mathbf{v}^T$ or $\mathbf{M}^T$, respectively. We denote by $||\mathbf{v}||$ the Euclidean norm of a vector $\mathbf{v}$, and by $||\mathbf{v}||_\infty$ its infinity norm. All logarithms are base 2. In this work, we mainly consider the uniform distribution and the centered discrete Gaussian distribution. For a finite set $S$ we associate $s \xleftarrow{\$} \mathcal{U}(S)$ to sample the element $s$ uniformly from $S$ (sometimes we simply write $s \xleftarrow{\$} S$). The centered discrete Gaussian distribution for $x \in \mathbb{Z}$ is defined to be $\mathcal{D}_\sigma = \rho_\sigma(x)/\rho_\sigma(\mathbb{Z})$, where $\sigma > 0$, $\rho_\sigma(x) = \exp(\frac{-x^2}{2\sigma^2})$, and $\rho_\sigma(\mathbb{Z}) = 1 + 2\sum_{x=1}^{\infty} \rho_\sigma(x)$. We denote by $d \xleftarrow{\$} \mathcal{D}_\sigma$ sampling a value $d$ randomly according to the distribution $\mathcal{D}_\sigma$.

Following the notation of [10], we define the rounding operator $\lfloor \cdot \rceil_d$ for some $d \in \mathbb{N}$ to be $\lfloor \cdot \rceil_d : \mathbb{Z} \to \mathbb{Z}, c \mapsto (c - [c]_{2^d})/2^d$ whereas $[c]_{2^d}$ denotes the unique integer in the set $(-2^{d-1}, 2^{d-1}] \subset \mathbb{Z}$ such that $c = [c]_{2^d} (\mod 2^d)$. The definition is easily extended to vectors by applying $\lfloor \cdot \rceil_d$ for each component. To apply the operator $\lfloor \cdot \rceil_d$ means essentially to drop the $d$ least significant bits.

[4] This technique was also used by Abdalla et al. [1], leading to the only other Fiat-Shamir lattice-based signature with tight reduction. Unfortunately, this imposes further conditions on their parameters yielding a less efficient signature scheme than the original scheme by Lyubashevsky [51].

[5] There are only two other lattice-based signature schemes proven secure in QROM which are GPV [37] and a variant of Lyubashevsky's scheme [52] proven secure by Dagdelen et al. [29].

We write $\lfloor v \rceil_d = \lfloor w \rceil_d \pmod q$ in order to say that we consider the equation $\lfloor v \pmod q \rceil_d = \lfloor w \pmod q \rceil_d$.

A function is called *negligible* in the security parameter $\lambda$, denoted by $negl(\lambda)$, if it decreases faster than the inverse of every polynomial in $\lambda$, for sufficiently large $\lambda$. For an algorithm $\mathcal{A}$, the value $y \leftarrow \mathcal{A}(x)$ denotes the output of $\mathcal{A}$ on input $x$; if $\mathcal{A}$ uses randomness then $\mathcal{A}(x)$ is a random variable. Also, $\mathcal{A}^{\mathcal{O}}$ denotes that $\mathcal{A}$ has access to oracle $\mathcal{O}$. An algorithm $\mathcal{A}$ is in probabilistic polynomial-time (PPT) if $\mathcal{A}$ is randomized — uses internal random coins — and, for any input $x \in \{0,1\}^*$, the computation of $\mathcal{A}(x)$ terminates in at most $poly(|x|)$ steps. A problem is called hard if there exist no polynomial time algorithm which solves the problem. In Appendix A we recall basic definitions and notations concerning a signature scheme and some definitions and facts about (q-ary) lattices.

## 2.2 The Learning with Errors Problem

In the following we recall the learning with errors (LWE) problem. In Appendix A.1 we also give the problem statement of the smallest integer solution (SIS) problem.

**Definition 1 (Learning with Errors Distribution).** *Let $n, m, q > 0$ be integers, $\mathbf{s} \in \mathbb{Z}_q^n$, and $\chi$ be a distribution over $\mathbb{Z}$. We define by $\mathcal{D}_{\mathbf{s},\chi}$ the LWE distribution which outputs $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$ and $e \leftarrow \chi$.*

Since our signature scheme is based on decisional LWE problem we omit the definition of the search version.

**Definition 2 (Learning with Errors Problem).** *Let $n, m, q > 0$ be integers and $\chi$ be a distribution over $\mathbb{Z}$. Moreover, define $\mathcal{O}_\chi$ to be an oracle, which upon input vector $\mathbf{s} \in \mathbb{Z}_q^n$ returns samples from the distribution $\mathcal{D}_{\mathbf{s},\chi}$. The decisional learning with errors problem $\mathsf{LWE}_{n,m,q,\chi}$ is $(t, \varepsilon)$-hard if for any algorithm $\mathcal{A}$, running in time $t$ and making at most $m$ queries to its oracle, we have*

$$\left| \Pr\left[\mathcal{A}^{\mathcal{O}_\chi(\mathbf{s})}(\cdot) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)}(\cdot) = 1\right] \right| \leq \varepsilon \ ,$$

*where the probabilities are taken over $s \xleftarrow{\$} \mathcal{U}(\mathbb{Z}_q^n)$ and the random choice of the distribution $\mathcal{D}_{\mathbf{s},\chi}$, as well as the random coins of $\mathcal{A}$.*

We note that the hardness of LWE is retained even if the secret vector $\mathbf{s}$ is sampled according to the error distribution $\chi$, known as the "normal form" [7,55]. We use the notation $\mathsf{LWE}_{n,m,q,\sigma}$ if $\chi$ is distributed according to $\mathcal{D}_\sigma$.

The LWE assumption comes with a worst-to-average-case reduction [22,58,60]; breaking certain average instances of LWE allows one to break all instances of certain standard lattice problems (namely GapSVP and SIVP).

## 2.3 Tightness

Let $\mathit{\Pi}$ be a cryptographic scheme with the security based on a hard problem $\mathcal{P}$, e.g., SIS or LWE. Let $\mathcal{A}$ be an algorithm which breaks the security of the scheme $\mathit{\Pi}$ — with respect to a security model — in time $t_\mathcal{A}$, and with a success probability of $\varepsilon_\mathcal{A}$. Let $\mathcal{R}$ be an algorithm, also called reduction, which solves the underlying problem $\mathcal{P}$ in time $t_\mathcal{R}$ with success probability $\varepsilon_\mathcal{R}$ by internally running the algorithm $\mathcal{A}$ (in a black-box way). We say the reduction of $\mathcal{P}$ to $\mathit{\Pi}$ is *tight* if $\varepsilon_\mathcal{A} \approx \varepsilon_\mathcal{R}$ and $t_\mathcal{A} \approx t_\mathcal{R}$. Otherwise, we call the reduction *loose* or *non-tight*. The term $(t_\mathcal{R} \cdot \varepsilon_\mathcal{A})/(t_\mathcal{A} \cdot \varepsilon_\mathcal{R})$ denotes the tightness gap. We call a problem $\mathcal{P}$ *n-bit hard* if $t_\mathcal{R}/\varepsilon_\mathcal{R} \geq 2^n$, and a scheme $\mathit{\Pi}$ *m-bit secure* if $t_\mathcal{A}/\varepsilon_\mathcal{A} \geq 2^m$. Note that a scheme $\mathit{\Pi}$ is not necessarily $n$-bit secure if its security is reduced to an $n$-bit hard problem $P$ — in particular, if the given reduction is non-tight.

Public Parameters:
$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n};\ m,n \in \mathbb{Z}, m > n;\ \sigma,\kappa,q \in \mathbb{N};\ H : \{0,1\}^* \to \{0,1\}^\kappa;\ F : \{0,1\}^\kappa \to \mathcal{B}_{n,\omega}$

```
Algorithm KeyGen
INPUT: $\mathbf{A}, n, m, q, \sigma$
OUTPUT: $(\mathsf{sk},\mathsf{vk}) = ((\mathbf{S},\mathbf{E}),\mathbf{T})$

  1. $\mathbf{S} \xleftarrow{\$} D_\sigma^{n\times n}$
  2. $\mathbf{E} \xleftarrow{\$} D_\sigma^{m\times n}$
  3. if checkE$(\mathbf{E}) = 0$
       then Restart
  4. $\mathbf{k} \xleftarrow{\$} \{0,1\}^\kappa$
  5. $\mathbf{T} \leftarrow \mathbf{AS} + \mathbf{E} \pmod q$
  6. $\mathsf{sk} \leftarrow (\mathbf{S},\mathbf{E}), \mathsf{vk} \leftarrow (\mathbf{T})$
  7. return $(\mathsf{sk},\mathsf{vk})$
```

```
Algorithm Sign
INPUT: $q, \mu, \mathbf{A}, \mathbf{S}, \mathbf{E}$,
OUTPUT: $(\mathbf{z}, c)$

1. $\mathbf{y} \xleftarrow{\$} [-B, B]^n$
2. $\mathbf{v} \leftarrow \mathbf{Ay} \pmod q$
3. $c \leftarrow H(\lfloor \mathbf{v} \rceil_d, \mu)$
4. $\mathbf{c} \leftarrow F(c)$
5. $\mathbf{z} \leftarrow \mathbf{y} + \mathbf{Sc}$
6. $\mathbf{w} \leftarrow \mathbf{v} - \mathbf{Ec} \pmod q$
7. if $|[\mathbf{w}_i]_{2^d}| > 2^{d-1} - L$,
   or $||\mathbf{z}||_\infty > B - U$
     then Restart
8. return $(\mathbf{z}, c)$
```

```
Algorithm Verify
INPUT: $q, \mu, \mathbf{z}, c, \mathbf{A}, \mathbf{T}$
OUTPUT: {0,1}

1. $\mathbf{c} \leftarrow F(c)$
2. $\mathbf{w}' \leftarrow \mathbf{Az} - \mathbf{Tc} \pmod q$
3. $c' \leftarrow H(\lfloor \mathbf{w}' \rceil_d, \mu)$
4. if $c' = c$ and $||\mathbf{z}||_\infty \le B - U$
     then return 1
5. return 0
```

Fig. 1: Signature scheme TESLA; for the implementation of checkE see [65]

## 3 The Signature Scheme TESLA

In this section, we present the lattice-based signature scheme TESLA which we prove tightly secure and demonstrate its high efficiency. While its construction was originally proposed by Bai and Galbraith [10] and later revisited by Dagdelen et al. [65] we are able to enhance its security, minimize the underlying assumptions, and improve the performance even further. Moreover, in our security reduction we get rid of the Forking Lemma which is in general an obstacle when proving quantum security [20, 29].

We (re-)name the signature scheme by Bai and Galbraith with its modifications by Dagdelen et al. to emphasize the various properties which we show in this paper. Throughout this paper we call it TESLA (Tightly-secure, Efficient signature scheme from Standard LAttices).

### 3.1 Description of the Signature Scheme TESLA

For easy reference the signature scheme TESLA = (KeyGen, Sign, Verify) is depicted in Figure 1. The concrete parameter set we propose can be found in Table 1 (and its derivation in Section 4).

**Public Parameters.** The scheme depends on the parameters $n$, $m$, $l$, $\alpha$, $\omega$, $d$, $B$, $q$, $U$, $L$, $\kappa$, and the distribution $\mathcal{D}_\sigma$. Let $m, n \in \mathbb{N}_{>0}$ with $m > n > \kappa$ and denote the standard deviation as $\sigma$. The remaining parameters are computed from $m$, $n$, $\kappa$, $\alpha$, and $\sigma$ as shown in Table 1. Let $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ be a uniformly random sampled matrix which is publicly known as a global constant and can be shared among arbitrary many signers.
The algorithms make use of a hash function $H(\cdot)$ which maps a bit string of arbitrary length to a bit string of length $\kappa$[6]. Furthermore, there is an encoding function $F : \{0,1\}^\kappa \to \mathcal{B}_{n,\omega}$ which takes the binary output of the hash function and produces a vector of length $n$ and weight $\omega$. For more information about the encoding function see [39].

**Key Generation.** At first, secret matrices $\mathbf{S} \in \mathbb{Z}^{n\times n}$ and $\mathbf{E} \in \mathbb{Z}^{m\times n}$ are sampled from the discrete Gaussian distributions $D_\sigma^{n\times n}$ and $D_\sigma^{m\times n}$, respectively. The matrix $\mathbf{E}$ has to satisfy certain constraints to ensure that the signatures are correct and short. These constraints are checked by the function CheckE which is introduced by Dagdelen et al. [65]. The check algorithm works as follows: for a matrix $\mathbf{E}$, define $\mathbf{E_h}$ to be the $h$-th row of $\mathbf{E}$. The function

[6] As it is common for signatures derived by the Fiat-Shamir transform, we instantiate a signature of bit security $\lambda$ using a random oracle which outputs $\lambda$ bits. As in [52], finding a collisions in the random oracle does not constitute a break.

$max_k(\cdot)$ returns the $k$-th largest entry of a vector. The key pair is rejected if for any row of $\mathbf{E}$ it holds that $\sum_{k=1}^{\omega} max_k(\mathbf{E_h})$ is greater then some bound $L$. Finally, the signing key $\mathsf{sk} = (\mathbf{S}, \mathbf{E})$ and public verification key $\mathsf{vk} = \mathbf{T} = \mathbf{AS} + \mathbf{E}$ are returned.

**Signing Algorithm.** First, sample the vector $\mathbf{y}$ uniformly random in $[-B, B]^n$. Afterwards, the higher order bits of $\mathbf{v} = \mathbf{Ay} \pmod q$ are hashed together with the message $\mu$ yielding the hash value $c$. Applying the encoding function to $c$ we obtain a value $\mathbf{c} = F(c)$. Further on, we compute $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$ in $\mathbb{Z}$. Now, rejection sampling is applied to make sure that the signature does not leak any information about the secret $\mathbf{S}$ and that the signature verifies for the applied methods for compressing. That is, if either $|[\mathbf{w}_i]_{2^d}| > 2^{d-1} - L$ or $||\mathbf{z}||_\infty > B - U$, with $\mathbf{w} = \mathbf{v} - \mathbf{Ec} \pmod q$, then the signing algorithm discards $(\mathbf{z}, c)$ and repeats all steps. Finally, it returns the signature $(\mathbf{z}, c)$ on the message $\mu$.

**Verification Algorithm.** The algorithm upon input a message $\mu$ and a signature $(\mathbf{z}, c)$, first computes $\mathbf{c} = F(c)$ to obtain $\mathbf{w}' = \mathbf{Az} - \mathbf{Tc} \pmod q$, and returns 1 if $c = H(\lfloor \mathbf{w}' \rceil_d, \mu)$ and $||\mathbf{z}||_\infty \leq B - U$ are both satisfied; otherwise, it returns 0.

## 3.2 Security Reduction

Bai and Galbraith [10] prove the security of their signature scheme assuming the hardness of *both* LWE and (unbalanced) SIS. The proof follows the standard way of proving Fiat-Shamir-type signatures, namely using the Forking Lemma proposed by Pointcheval and Stern [59]. As mentioned earlier, although the Forking Lemma is a powerful and actively-used tool to prove security of signatures, the obtained security reductions are not tight. Furthermore, the use of the Forking Lemma makes it hard to prove security of schemes against quantum adversaries.

To avoid this lemma we use a reduction method introduced by Katz and Wang [42]. The underlying idea is to give a hypothetical adversary against the signature scheme either a genuine-generated public key of the system or a fake – i.e., random – one. The underlying assumption is that those keys cannot be distinguished easily. Now, if the scheme guarantees that for fake public keys the existence of valid signatures is statistically bounded by a negligible probability, the hypothetical adversary will simply fail to forge in case a fake public key is given. On the other hand, in case a public key is generated honestly, the adversary will output a forgery by assumption. This different behavior of the adversary helps the security reduction to distinguish between the two samples.

The following theorem shows that the signature scheme TESLA is unforgeable. Specifically, our security reduction is tight and the security of TESLA relies solely on the decisional learning with errors problem in the random oracle model.

**Theorem 1.** *Let the parameters $n, m, \omega, d, B, q, U, L, \sigma$ be arbitrary but satisfying the constraints in Table 1. If $\mathsf{LWE}_{n,m,q,\sigma}$ is $(t_\mathcal{D}, \varepsilon_\mathcal{D})$-hard, the signature scheme TESLA in Figure 1 is $(t_\mathcal{A}, q_h, q_s, \varepsilon_\mathcal{A})$-unforgeable against adaptively chosen-message attacks in the random oracle model where $t_\mathcal{D} \approx t_\mathcal{A} + \mathcal{O}(q_s\kappa^3)$ and*

$$\varepsilon_\mathcal{D} \approx \varepsilon_\mathcal{A}\left(1 - \frac{q_s(q_s+q_h)2^{(d+1)\cdot m}}{(2B+1)^n q^{m-n}}\right) - \frac{q_h 2^{d\cdot n}(2B-2U+1)^n}{q^m} - \frac{(28\sigma+1)^{m\cdot n+n^2}}{q^{m\cdot n}}.$$

Before proving our main theorem we build some supplementary results. First, we recall a useful lemma stated by Bai and Galbraith [10, Lemma 3] which gives us information about the number of possible values for $\lfloor \mathbf{Ay} \pmod q \rceil_d$.

**Lemma 1.** *Let the parameters $n, m, d, B,$ and $q$ be as in Theorem 1. Furthermore, let $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$. Then we have,*

$$\Pr\left[\lfloor \mathbf{Ay_1} \rceil_d = \lfloor \mathbf{Ay_2} \rceil_d \pmod q \mid \mathbf{y_1}, \mathbf{y_2} \xleftarrow{\$} [-B, B]^n\right] \leq \frac{2^{(d+1)\cdot m}}{(2B+1)^n \cdot q^{m-n}}.$$

In our security reduction we show that if an algorithm $\mathcal{A}$ is given a randomly chosen tuple $(\mathbf{A},\mathbf{T})$, then there exists a valid signature only with negligible probability. To this end, we utilize the following lemmata. The proof of Lemma 2 and Lemma 3 can be found in Appendix B.

**Lemma 2.** *Let the parameters $n$, $m$, $d$, $B$, and $q$ be as in Theorem 1. Let $\mathbf{A},\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$. Furthermore, let $\mathbf{y} \xleftarrow{\$} [-B,B]^n$, $\mathbf{c} \xleftarrow{\$} \mathcal{B}_{n,\omega}$, and $\delta \in \mathbb{Q}_{>0}$. Then, it holds that*

$$\Pr\left[\ \exists\ \mathbf{z} \in [-\delta,\delta]^n \mid \lfloor \mathbf{Az} \rceil_d = \lfloor \mathbf{Ay}+\mathbf{Tc} \rceil_d \pmod q\ \right] \leq \frac{2^{d\cdot n}\cdot(2\delta+1)^n}{q^m}\ ,$$

*where the probability is taken over random choices of $\mathbf{A}$ and $\mathbf{T}$.*

We reformulate the lemma and obtain the following corollary useful for the proof of our main theorem.

**Corollary 1.** *Let $\mathbf{A}$, $\mathbf{T}$, $\mathbf{y}$, $\mathbf{c}$, $\delta$, $q$, $n$, $m$, $B$, and $d$ be defined as in Theorem 1. Then, it holds that*

$$\Pr\left[\ \exists\ \mathbf{z} \in [-\delta,\delta]^n \mid \lfloor \mathbf{Ay} \rceil_d = \lfloor \mathbf{Az}-\mathbf{Tc} \rceil_d \pmod q\ \right] \leq \frac{2^{d\cdot n}\cdot(2\delta+1)^n}{q^m},$$

*where the probability is taken over random choices of $\mathbf{A}$ and $\mathbf{T}$.*

In the security reduction we use that for randomly chosen matrices $\mathbf{A},\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, where $n,m$ are chosen with respect to the security parameter $\lambda$, the probability that there exist matrices $\mathbf{S}$ and $\mathbf{E}$ with "small" entries such that $\mathbf{AS}+\mathbf{E}=\mathbf{T}$ is negligible in $\lambda$. This statement is captured in the following lemma.

**Lemma 3.** *Let $n$, $m$, $q$, and $\sigma$ be chosen as defined in Theorem 1. Furthermore, let $\mathbf{A},\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$. The probability that there exist matrices $\mathbf{S} \in [-k\sigma,k\sigma]^{n\times n}$ and $\mathbf{E} \in [-k\sigma,k\sigma]^{m\times n}$, for some $k\in\mathbb{N}$, such that $\mathbf{AS}+\mathbf{E}=\mathbf{T}$, is bounded by the following term*

$$\Pr\left[\ \exists\ \mathbf{S} \in [-k\sigma,k\sigma]^{n\times n}, \mathbf{E} \in [-k\sigma,k\sigma]^{m\times n} \mid \mathbf{AS}+\mathbf{E}=\mathbf{T}\ \right] \leq \frac{(2k\sigma+1)^{m\cdot n+n^2}}{q^{m\cdot n}}\ .$$

We are now able to prove Theorem 1.

*Proof (Theorem 1).*
Let $\mathcal{A}$ be an algorithm which runs in time $t_{\mathcal{A}}$, makes $q_h$ hash queries and $q_s$ sign queries, and forges a signature with probability $\varepsilon_{\mathcal{A}}$. We show how to build a distinguisher $\mathcal{D}$ solving $\mathsf{LWE}_{n,m,q,\sigma}$ in time $t_{\mathcal{D}}$ with probability $\varepsilon_{\mathcal{D}}$ as stated in the theorem.

Algorithm $\mathcal{D}$ upon input tuple $(\mathbf{A},\mathbf{T})$ has to decide whether $\mathbf{T}$ is a matrix sampled uniformly from $\mathbb{Z}_q^{m\times n}$ or whether it is of the form $\mathbf{T}=\mathbf{AS}+\mathbf{E}$ for some $\mathbf{S} \xleftarrow{\$} D_\sigma^{n\times n}$ and $\mathbf{E} \xleftarrow{\$} D_\sigma^{m\times n}$. That means, $\mathcal{D}$ outputs 1 if $\mathbf{T}=\mathbf{AS}+\mathbf{E}$ and 0 otherwise. The algorithm $\mathcal{D}$ uses $\mathcal{A}$ as a black-box. Algorithm $\mathcal{A}$ expects as input the public key. To this end, $\mathcal{D}$ hands over its own challenge tuple $(\mathbf{A},\mathbf{T})$. The responses to the hash and sign queries made by $\mathcal{A}$ are simulated as follows:

**Hash queries:** Algorithm $\mathcal{D}$ answers with values $c$ uniformly sampled from $\{0,1\}^\kappa$; however, if an input to the oracle repeats, we keep being consistent and reply with the same hash value as before.

**Sign queries:** Upon input a message $\mu$, $\mathcal{D}$ simulates a signature $(\mathbf{z},c)$ on $\mu$ by the following steps: $\mathcal{D}$ chooses uniformly random $c \xleftarrow{\$} \{0,1\}^\kappa$ and $\mathbf{z} \xleftarrow{\$} [-B+U,B-U]^n$, computes $\mathbf{c}=F(c)$ and $\mathbf{w}=\mathbf{Az}-\mathbf{Tc} \pmod q$, and checks whether $|[\mathbf{w}_i]_{2^d}| < 2^{d-1}-L$ for all $i\in\{1,\cdots,m\}$. If the latter is not fulfilled, $\mathcal{D}$ chooses $c$ and $\mathbf{z}$ again and repeats. Moreover, if the oracle was queried before on that input, namely on $(\lfloor \mathbf{w} \rceil_d,\mu)$, then $\mathcal{D}$ aborts the simulation. Otherwise, $\mathcal{D}$ returns $(\mathbf{z},c)$.

Eventually, $\mathcal{A}$ outputs a forgery $(\tilde{\mathbf{z}}, \tilde{c})$ on some message $\tilde{\mu}$, which was not queried to the sign oracle. If $\mathsf{Verify}(\mathsf{vk}, \mu, (\tilde{\mathbf{z}}, \tilde{c})) = 1$, algorithm $\mathcal{D}$ returns 1; else $\mathcal{D}$ returns 0.

In the following, we distinguish between two cases where $\mathbf{T}$ follows the LWE distribution — i.e, $\mathbf{T} = \mathbf{AS} + \mathbf{E}$ — or $\mathbf{T}$ was sampled uniformly.

**1st case, $\mathbf{T} = \mathbf{AS} + \mathbf{E}$:** By [10, Lemma 4], we know that for the algorithm $\mathcal{A}$ the response to hash or sign queries from the simulation by $\mathcal{D}$ is indistinguishable from responses made by a real hash function and signer. Thus, the only possibility that $\mathcal{D}$ falsely outputs 0 is when $\mathcal{D}$ aborts during a sign query or when the algorithm $\mathcal{A}$ fails. The probability that $\mathcal{A}$ fails is $1 - \varepsilon_{\mathcal{A}}$. By Lemma 1, the probability that $\mathcal{D}$ will abort during the simulation of $\mathcal{A}$'s environment is $q_s(q_h + q_s) \cdot \frac{2^{(d+1)\cdot m}}{(2B+1)^n \cdot q^{m-n}}$. Thus, $\mathcal{D}$ outputs 1 correctly with probability at least $\varepsilon_{\mathcal{A}} \left(1 - q_s(q_h + q_s) \cdot \frac{2^{(d+1)\cdot m}}{(2B+1)^n \cdot q^{m-n}}\right)$.

**2nd case, $\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$:** First, we can bound the entries of matrices $\mathbf{S}$ and $\mathbf{E}$ with high probability by $14 \cdot \sigma$, since they are Gaussian distributed with standard deviation $\sigma$. We stress that by Lemma 3 the probability that there exist matrices $\mathbf{S} \in [-14\sigma, 14\sigma]^{n\times n}$ and $\mathbf{E} \in [-14\sigma, 14\sigma]^{m\times n}$ such that $\mathbf{T} = \mathbf{AS} + \mathbf{E}$ for $\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$ is smaller or equal $\frac{(2\cdot 14\cdot\sigma+1)^{m\cdot n+n^2}}{q^{m\cdot n}}$. Assume $\mathbf{T}$ is not of this form (which is the case with probability greater than $1-2^{\lambda}$ with our choice of parameters). We have to bound the probability that for $c \xleftarrow{\$} \mathcal{B}_{n,\omega}$ there exists a vector $\mathbf{z}$ with $||\mathbf{z}||_\infty \leq B-U$ such that $\lfloor \mathbf{Ay} \rceil_d = \lfloor \mathbf{Az} - \mathbf{Tc} \rceil_d \pmod q$. Let $c \in \mathcal{B}_{n,\omega}$ be fix but arbitrary. By Corollary 1, the probability that such a vector $\mathbf{z}$ exist is smaller than $\frac{2^{d\cdot n}\cdot(2(B-U)+1)^n}{q^m}$. We also stress that $\mathcal{A}$ cannot exploit the simulated signatures to generate a forgery, since any other message would yield a different random "challenge" value $c$. Only collisions in the random oracle may help algorithm $\mathcal{A}$ to forge a signature which, however, will only be found with negligible probability. Hence, we have an upper bound on the probability of $\mathcal{D}$ falsely returning 1 of $q_h \cdot \frac{2^{d\cdot n}(2(B-U)+1)^n}{q^m} + \frac{(28\sigma+1)^{m\cdot n+n^2}}{q^{m\cdot n}}$.

Finally,

$$\left|\Pr\left[\mathbf{S} \xleftarrow{\$} D_\sigma^{n\times n}, \mathbf{E} \xleftarrow{\$} D_\sigma^{m\times n} : \mathcal{D}(\mathbf{A}, \mathbf{AS}+\mathbf{E}) = 1\right] - \Pr\left[\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n} : \mathcal{D}(\mathbf{A}, \mathbf{T}) = 1\right]\right|$$
$$\geq \varepsilon_{\mathcal{A}} \cdot \left(1 - \frac{q_s\cdot(q_s+q_h)\cdot 2^{(d+1)\cdot m}}{(2B+1)^n\cdot q^{m-n}}\right) - \frac{q_h \cdot 2^{d\cdot n}\cdot(2B-2U+1)^n}{q^m} - \frac{(28\sigma+1)^{m\cdot n+n^2}}{q^{m\cdot n}} \geq \varepsilon_{\mathcal{D}}.$$

It remains to show that the time $t_{\mathcal{D}}$ is close to the running time of $\mathcal{A}$. The running time $t_{\mathcal{D}}$ of $\mathcal{D}$ includes the running time $t_{\mathcal{A}}$ of the algorithm $\mathcal{A}$. Besides, $t_{\mathcal{D}}$ is dominated by the computational time of answering sign queries during the simulation, which essentially consists of a constant (small) number of a matrix-vector multiplication. Such a multiplication runs in complexity $\mathcal{O}(\kappa^3)$. In every simulation of a signature query the probability that the simulated signature is returned, i.e., $|[\mathbf{w}_i]_{2^d}| < 2^{d-1} - L$, can be bounded by $\left(\frac{2^d - L}{2^d}\right)^m$. Because $\mathbf{A}$, $\mathbf{T}$, $c$, and $\mathbf{z}$ are chosen uniformly random, $\mathbf{w}$ is uniformly random distributed in $\mathbb{Z}_q^m$. Since $q$ is a large integer, also the $d-$least significant bits are distributed (closely to) uniformly random distributed in the range of $[-2^d, 2^d]$. Thus, the ratio of $2^d - L$ and $2^d$ gives the probability we are looking for. Since $L$ is always greater than 2, the probability that $|[\mathbf{w}_i]_{2^d}| < 2^{d-1} - L$ is always greater than 1/2. Therefore, we get $t_{\mathcal{D}} \approx t_{\mathcal{A}} + \mathcal{O}(q_s\kappa^3)$. □

*Remark 1 (Strongly unforgeability against chosen-message attacks.).* We remark that we prove TESLA unforgeable against chosen-message attacks. Alternatively, we could aim for a stronger security model, namely strong unforgeability. Roughly speaking, a signature scheme is strongly unforgeable if an adversary is not able to forge a new signature which she has not received from the signing oracle. In contrast to standard unforgeability, an adversary also succeeds if she finds a forgery for a message she has already queried to the signing oracle (as long as the signature is different). In order to prove TESLA strongly unforgeable we would need to be based on the SIS assumption to prevent an adversary to find collisions in the hash input. We note that other

**Table 1:** Parameter sets in comparison; sizes are given in mega byte [MB] or kilo byte [kb]

| Parameter selection | | | | |
|---|---|---|---|---|
| Parameter | Bound | Dagdelen et al. [65] | TESLA-128 | TESLA-256 |
| $\kappa$ | | 128 | 128 | 256 |
| $n$ | | 532 | 416 | 576 |
| $m$ | | 840 | 800 | 1056 |
| $\sigma$ | | 43 | 114 | 30 |
| $\alpha$ | | 128 | 1 | 1 |
| $l$ | | 3 | 2.65 | 2.8 |
| $L$ | $l \cdot \omega \cdot \sigma$ | 2322 | 6042 | 3547 |
| $\omega$ | $2^{\omega}\binom{n}{\omega} \geq 2^{\kappa}$ | 18 | 20 | 43 |
| $B$ | $\geq 14 \cdot (n-1) \cdot \sqrt{\omega}\sigma$ | $2^{21}-1$ | $2^{23}-1$ | $2^{22}-1$ |
| $U$ | $14 \cdot \sqrt{\omega}\sigma$ | 2554.1 | 7138 | 2754 |
| $d$ | $(1-2\cdot L/2^d)^m \geq 0.4$ | 23 | 24 | 24 |
| $q$ | $\geq \left(\frac{2^{(d+1)\cdot m+\alpha}}{(2\cdot B)^n}\right)^{1/(m-n)}$ $\geq 4 \cdot B$ | $2^{29}-3$ | $2^{27}-39$ | $2^{29}-3$ |
| Prob. of acc., KeyGen | empirically | 0.99 | 0.35 | 0.99 |
| Prob. of acc., Sign | | 0.314 | 0.357 | 0.408 |
| public-key size | $m \cdot n \cdot \lceil \log_2(q) \rceil$ | 1.54 MB | 1.33 MB | 2.20 MB |
| secret-key size | $(n^2 + n \cdot m)\lceil \log_2(14\sigma) \rceil$ | 0.87 MB | 01.01 MB | 1.06 MB |
| signature size | $n \cdot \lceil \log_2(2B) \rceil + \kappa$ | 11.68 kb | 10.24 kb | 13.50 kb |

lattice-based signature schemes are also proven (standard) unforgeable as it is seen sufficient for many applications.

## 4 Selecting Parameters for TESLA

In this section we propose parameters for TESLA such that the signature scheme is 128-bit secure in both the classical and post-quantum setting. In particular, we deduce parameters using state-of-the-art methods on assessing the concrete hardness of LWE against classical and against quantum adversaries.

### 4.1 Optimizing for Software Efficiency

We proceed similarly to Dagdelen et al. [65] when selecting parameters for TESLA. However, our security reduction for TESLA minimizes the underlying assumptions which allows us to choose secure parameters from a greater set of choices. More precisely, our parameters do not have to involve a 128-bit hard instance of the SIS assumption. In fact, our underlying SIS assumption is only 108-bit for TESLA-128 (see Table 2). Still, our parameters yield an 128-bit secure signature scheme.

Table 1 illustrates our concrete choice of parameters for 128-bit security. Note that we introduce a new parameter $\alpha$, which is not present in previous descriptions of the scheme [10,65]. This value gives an upper bound for the probability that a forger $\mathcal{A}$ in the security game can distinguish the simulated signatures from honestly generated ones in the security reduction (see Section 3.2). More precisely, the distinguishing probability is bounded by $\frac{q_s \cdot (q_s+q_h) \cdot 2^{(d+1)\cdot m}}{(2B+1)^n \cdot q^{m-n}}$. Bai and Galbraith [10] chose parameters, in particular the modulus $q$ such that the distinguishing probability is negligible in the security parameter, i.e., smaller than $2^{-\lambda}$. We stress however, that it suffices if the forger $\mathcal{A}$ cannot differentiate them in at least half of the (complete) runs. In that case, the reduction has to run $\mathcal{A}$ twice (expected) to succeed. Doing so, one has to take the loss of security due to this distinguishing probability into account. Nonetheless, if $\mathcal{A}$ succeeds to distinguish genuine from fake signatures in less than 50% of the cases, the signature instantiation merely looses one bit of security.

**Table 2:** Comparison of the security of TESLA with our parameter sets to the parameter set proposed by Dagdelen et al. [65].

| Problem | Attack | Bit Security | | |
|---|---|---|---|---|
| | | Dagdelen et al. [65] | TESLA-128 | TESLA-256 |
| LWE | Decoding | 276 | 132 | 135 |
| | Embedding | 134 | 131 | 132 |
| SIS | Lattice reduction | 157 | 108 | 175 |

Another important parameter of the signature scheme is the value $L$. In the original work [10], Bai and Galbraith set $L = 7\omega\sigma$, whereas Dagdelen et al. [65] chose $L = 3\omega\sigma$. We propose a different way to choose parameter $L$. That is, we select $L$ such that the function CHECKE excepts an error matrix $\mathbf{E}$ with probability higher than $2^{-2}$; in [10, 65] $\mathbf{E}$ is accepted with probability close to 1. This way we are able to select $L$ more aggressively. We note that the smaller the value $L$ is the higher the probability of acceptance in the signature algorithm (line 7, first part) becomes. The probability for acceptance is $(1-2L/2^d)^m$. Our choice of parameter yields an concrete acceptance probability in line 7, first part, of Sign for TESLA-128 of approximately 56.2% and for TESLA-256 of approximately 64%. Remarkably, for TESLA-256 we obtain a higher acceptance probability during the sign algorithm although we have a very high acceptance probability in CHECKE as well. Note that rejecting 65% of secret key instances lowers the bit security of TESLA-128 again by two bits. Overall, a signature $(c, \mathbf{z})$ is accepted with probability 35.7% (resp. 40.8%) which is larger than in [65].

In summary, our parameter yield a secret key (resp. public key) size of 1.01 MB (resp. 1.33 MB) for TESLA-128 and 1.06 MB (resp. 2.20 MB) for TESLA-256. These sizes come off since we rely on lattice problems on standard lattices without introducing any structure being potentially exploited in future. Our instantiation of TESLA-128 gives signature sizes of 10.24 kb which improves the proposal by [10] (albeit their shortest instantiation of 11.68 kb of signature size provides 83 bits of security as pointed out by [65]).

## 4.2 Concrete Bit Security of TESLA Against Classical Adversaries

To estimate the hardness of LWE we consider state-of-the-art lattice attacks. There are mainly two families of algorithms for solving LWE. First, there is the decoding attack which goes back to 1986 where Babai [9] proposed the nearest plane algorithm. Today, improved versions by Linder and Peikert [49] and Liu and Nguyen [50] are used. Here, the given lattice basis is first reduced by a lattice reduction algorithm (for instance, the BKZ algorithm [27]) and then given the nearest plane algorithm (or faster variants) as input to find the closest vector to a target vector.

The second approach is to convert the LWE instance to the (unique) shortest vector instance. This is called the embedding approach. There are two ways to define the underlying lattice for which the solution of the (unique) shortest vector instance is of interest. Depending on the number of given LWE samples the appropriate lattice is chosen. We refer to [65] for a formal description of the embedding attacks. The resulting bit security of both attacks is shown in Table 2.

Alternatively, one could use non-lattice algorithms to solve the LWE problem. There is, for instance, the algorithm by Blum, Kalai, and Wassermann [19]. It received a lot of attention in the last years [5, 6, 17, 35, 48]. During the algorithm run one tries to find a transformation matrix $\mathbf{B}$ to introduce more structure to the LWE matrix $\mathbf{A}$. To this end, one needs a large number of LWE samples. Although the number of samples necessary was crucially reduced by Duc et al. [31], it is still not applicable (by far) for our instances. The same drawback exists in the algorithm by Arora and Ge [8]. The main idea here is to transform a noisy linear system of equations to a noise-free non-linear system of equations, via linearization [8] or via Gröbner bases [4]. Again, their approach requires far more LWE samples than given in our LWE instance. For this reason, their runtimes are not considered here. We note that asymptotically those algorithms are the fastest algorithms to solve LWE but not yet applicable to current LWE-based signatures.

### 4.3 Concrete Bit Security of TESLA Against Quantum Adversaries

To the best of our knowledge, there is no quantum algorithm known to solve learning with errors directly. Instead quantum speed ups on the building blocks of aforementioned attacks are investigated. State-of-the-art quantum attacks on lattice-based problems mainly make (black-box) use of Grover's quantum search algorithm [38] to speed up classical algorithms. Laarhoven et al. [47] investigate and compare the impact of Grover's quantum search algorithm on different solvers for the shortest vector problem (SVP). There are three main approaches to solve SVP: probabilistic sieving algorithms [3], Voronoi cell computations [57], and enumeration algorithms [61]. For an extensive overview of the different algorithms and state-of-the-art solvers, see [47]. Probabilistic sieving algorithms and Voronoi cell algorithms run on single exponential time, but experiments show that enumeration algorithms are faster on currently used instances. Especially, Voronoi cell algorithms are not practical all together for our instances, hence we only consider enumeration and sieving algorithms in the following.

To the best of our knowledge the current fastest heuristic for the runtime (in seconds) of sieving algorithms for high lattice dimensions $n$ was recently given by Laarhoven and Weger [46] as

$$time_{sieve} = 2^{0.2972n+o(1)}. \tag{1}$$

The current fastest enumeration implementation is due to Kuo et al. [45][7]. Kuo et al.'s implementation observes a runtime (in seconds) of

$$time_{enum} = 2^{0.00059n^2+0.11n-5.8} \text{ if } n \geq 112. \tag{2}$$

Figure 2 shows a comparison of the runtimes given in Equations (1) and (2) in seconds. We stress that we display the various runtimes without considering their constant overhead. Due to the non-existence of quantum computers (as far as we know) which could perform those computations in practice, we cannot yet determine the constant term in any runtime prediction of quantum SVP solvers. We are aware that such constants play an important role in general, but for our purpose of comparison the constant overhead has no impact as enumeration algorithms in practice perform significantly better in relevant lattice dimensions. As can be seen in Figure 2, sieving algorithms start outperforming enumeration algorithms with an approximate dimension of 320.

Considering quantum lattice attacks, Laarhoven et al. [47] state that the application of Grover's quantum search algorithm to Voronoi cell computations and enumeration algorithms does not lead easily to speed ups. In contrast it has a serious impact on sieving algorithms as Laarhoven et al. [47] propose a new, faster SVP solver with a runtime (in seconds) of

$$time_{\text{q-sieve}}(n) = 2^{0.286n+o(1)}. \tag{3}$$

As shown in Figure 2 the quantum sieving algorithm by Laarhoven et al. [47] is faster than enumeration algorithms, starting from an approximate dimension of 300. We observe that until dimension roughly 300 enumeration algorithms perform better than both classical and quantum sieving algorithms.

In most efficient attacks SVP solvers are rather used on sublattices of lower dimension, as in the BKZ algorithm [28], also called blocksize, than on the lattice defined by an LWE matrix. Due to Nguyen and Chen [28] a block size of about 200 is sufficient to use for 128 bit of security. Running BKZ with a blocksize of over 300 would be an overkill for the targeted security level, and would be only relevant for security levels of over 200 bits.

We conclude that today's quantum attacks on LWE are only preferable for very high dimension and are not relevant to our instantiations. However, we believe that there is a big gap in the literature in investigating quantum hardness of lattice problems in general. We see a huge potential

[7] We note that their implementation does not make use of the fastest building blocks known today, such as BKZ 2.0 [27]. Still, no runtime estimates are publicly accessible of faster enumeration algorithms. For our purpose Kuo et al. [45] is fast enough to outperform quantum sieving algorithms for our lattice targets.

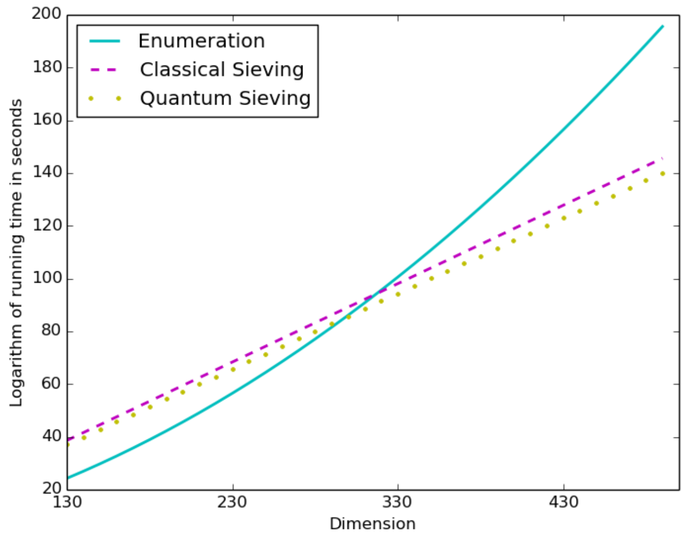

Fig. 2: An extrapolated-runtime comparison (in seconds) of the state-of-the-art implementations of the enumeration method [45] and both the classical sieving algorithm [46] and the quantumly enhanced variant [47] (asymptotically).

in this research field and thus pose the challenge to find better quantum algorithms on LWE that would make a revision of our parameters necessary for 128-bits of security.

Nevertheless, the bit security of TESLA does not only depend on the bit hardness of LWE but also on the security of the used hash function. As stated by Bernstein et al. [15] we have to increase the output length of the hash function from $\kappa$ to $2\kappa$ to receive a bit security of $\kappa$.

## 5 Software Implementation

To demonstrate the efficiency of our proposed parameter set we present a software implementation targeting the Intel Haswell microarchitecture. The starting point for our implementation is the software presented by Dagdelen et al., which we obtained from the authors. Just like the software presented in [65], also our software is systematically protected against timing attacks. The software makes use of the fast AVX2 instructions on vectors of 4 double-precision floating-point numbers. The reason that signing and verification is faster with the parameters proposed for TESLA might be obvious from the fact that $n$, $m$, and $q$ are smaller than the parameters proposed by Dagdelen et al [65]. However, signing with our parameters needs slightly more attempts to find a valid signature and the slowdown from increased number of average iterations might be larger than the effect from smaller cost per iteration. The following two modifications to the software proposed by Dagdelen et al. address this issue.

*Parallel Matrix-Vector Multiplication.* The most costly operation during signing is the computation of $\mathbf{Ay}$, which requires $n \cdot m$ multiplications in $\mathbb{Z}_q$, most of those followed by accumulation. Intel Haswell processors can perform two multiply-accumulate instructions on 256-bit vectors of double-precision floating point numbers every cycle. As we represent elements of $\mathbb{Z}_q$ as double-precision floating point numbers, one obtains a lower bound of $n \cdot m/8$ cycles per matrix multiplication. This lower bound corresponds to 41600 cycles for TESLA-128 and to 65463 cycles for TESLA-256.

Already Dagdelen et al. [65] pointed out that their actual performance is much lower. The reason is that each coefficient from $\mathbf{A}$ needs to be loaded from (at best) L2 cache, because the whole matrix $\mathbf{A}$ does not fit into the 32KB of K1 cache. The actual performance of matrix-vector multiplication presented by Dagdelen et al. was a factor of 5 slower than the lower bound.

However, on average, signing computes multiple of those matrix-vector multiplications, all with the constant matrix $\mathbf{A}$. Our software always samples $k = 4$ vectors $y$, then performs 4 matrix-vector multiplications and then proceeds to investigate whether one of the 4 results is usable for the signature. The disadvantage of this technique is that most of the time we compute some matrix-vector products that are not used in the end. The advantage is that matrix coefficients, once loaded from L2 cache are used six times instead of just once. The optimal value of $k$ depends on the parameter set. We wrote scripts that generate an optimized assembly routine for different parameters and different values of $k$. With those scripts we generated and benchmarked code for many different combinations that all offered the targeted security and then picked the fastest one for TESLA-128 and TESLA-256.

Note that signature verification cannot use the $k$-times-parallel matrix-vector multiplication for the computation of $\mathbf{Az}$. For this task we use an approach similar to [65].

*Lazy Reductions.* The coefficients of the matrix $\mathbf{A}$ are in the interval $(-\lfloor q/2 \rfloor, \lceil q/2 \rceil]$ and coefficients of $\mathbf{y}$ are in the interval $[-B, B]$. For our parameter choices of $q$ and $B$ in TESLA-128, each product of coefficients in the matrix vector multiplication has up to 49 bits. The mantissa of a double-precision floating-point value has 53 bits, so when accumulating $n = 416$ such products in the matrix-vector multiplication we need to reduce modulo $q$ several times. This was no different for the software by Dagdelen et al. [65], which is "overly conservative" and reduces after 7 multiply-accumulates. We reduce only after 16 multiply-accumulates. In TESLA-256 the products reach up to 52 bits and so we have to reduce after each addition; these frequent reductions account for a large portion of the performance difference between TESLA-128 and TESLA-256.

## 6 Results and Comparison

Table 3 gives benchmarking results of TESLA-128 and TESLA-256 and compares those benchmarks to state-of-the art results from the literature. As indicated in the table, we obtain all our benchmarks on an Intel Core-i7 4770K (Haswell) processor. We followed the standard practice of disabling Turbo Boost and hyperthreading. Benchmarks of TESLA for signing are averaged over $100,000$ signatures; benchmarks of TESLA for verification are the median of 100 verifications. The reason for not reporting the median for TESLA signing performance is that because of the rejection sampling, it would be overly optimistic.

Both our TESLA-128 software and the software presented in [65] use the same construction and target the 128-bit pre-quantum security level. There are two reasons that our software is almost two times faster on the same microarchitecture: First, the reduction to LWE (instead of LWE and SIS) allows us to choose more efficient parameters. Secondly, the parallel matrix-vector multiplication technique offers additional speedup.

The two ideal-lattice-based schemes listed in Table 3 are still faster than TESLA. However, the GLP software from [40] offers less than 80 bits of security and all available BLISS software leaks timing information in the Gaussian sampling, which is unavoidable for signing (or at least expensive). We expect that serious optimization of BLISS software can outperform the current results, but it would be very interesting to see how large the penalty is for a timing-attack-protected implementation of BLISS.

In the realm of yet small world of signatures that offer 128 bits of post-quantum security, TESLA-256 offers an interesting alternative to SPHINCS. Public and secret keys of TESLA-256 are much larger than SPHINCS keys, but TESLA-256 is much faster for signing, significantly faster for verification, and signatures are more than an order of magnitude smaller.

The post-quantum multivariate-based signature scheme Rainbow5640 [26, 30] performs best among all listed schemes but unfortunately, comes with no security reduction to its underlying problem.

**Table 3:** Overview of state-of-the-art post-quantum signature schemes. The column "ROM?, tight?" states whether the scheme has a security reduction in the random oracle model and whether this reduction is tight; "QROM?, tight?" states the same for the quantum random oracle model; "Security (PreQ)" lists the claimed pre-quantum security level; "Security (PostQ)" lists the claimed post-quantum security level, if available; "TAP?" indicates whether the software is protected against timing attacks.

| Scheme/Software | Comp. Assumptions | ROM?, Tight? | QROM?, Tight? | Security (PreQ) | Security (PostQ) | CPU | TAP? | Size (bytes) | cycles |
|---|---|---|---|---|---|---|---|---|---|
| **Selected signature schemes over standard lattices** | | | | | | | | | |
| GPV [11, 37] | SIS | yes, yes | yes, yes | 100 | ? | AMD Opteron 8356 (Barcelona) | no | vk: 157,286,400<br>sk: 81,788,928<br>sig: 200,704 | sign: $71,300,000$<br>verify: $9,200,000$ |
| BG [10, 65] | SIS, LWE | yes, no | - | 128 | ? | Intel Core i7-4770K (Haswell) | yes | vk: 25,406,610<br>sk: 7,298,048<br>sig: 1,495 | sign: $1,203,924$<br>verify: $335,072$ |
| TESLA-128 (this paper) | LWE | yes, yes | no[a] | 128 | ? | Intel Core-i7-4770K (Haswell) | yes | vk: $1,331,200$<br>sk: $1,011,712$<br>sig: $1,280$ | sign: $668,822$<br>verify: $233,420$ |
| TESLA-256 (this paper) | LWE | yes, yes | no[a] | $> 128$ | 128 | Intel Core-i7-4770K (Haswell) | yes | vk: $2,204,928$<br>sk: $1,057,536$<br>sig: $1,688$ | sign: $1,287,268$<br>verify: $508,285$ |
| **Selected signatures schemes over ideal lattices** | | | | | | | | | |
| GLP [39, 40, 65][b] | DCK | yes, no | - | $75 - 80$ | ? | Intel Core i5-3210M (Ivy Bridge) | yes | vk: 1,536<br>sk: 256<br>sig: 1,186 | sign: $452,223$<br>verify: $34,004$ |
| BLISS-BI [33, 34][c] | R-SIS, NTRU | yes, no | - | 128 | ? | "Intel Core 3.4GHz" | no | vk: 7,168<br>sk: 2,048<br>sig: 1,559 | sign: $\approx 351,333$<br>verify: $102,000$ |
| **Selected other post-quantum signature schemes** | | | | | | | | | |
| SPHINCS-256 [15] | Red. to hash collisions, tight<br>Red. to 2nd preimages, not tight<br>Both in the standard model | | | $> 128$ | 128 | Intel Xeon E3-1275 (Haswell) | yes | vk: $1,056$<br>sk: $10,88$<br>sig: $41,000$ | sign: $51,636,372$<br>verify: $1,451,004$ |
| Rainbow5640 [26, 30] | MQ, EIP[e] | - | - | 80 | ? | Intel Xeon E3-1275 (Haswell) | ? | vk: $44,160$<br>sk: $86,240$<br>sig: 37 | sign: $42,700$[d]<br>verify: $36,072$[d] |

[a] The auxiliary material of this paper gives a tight security reduction of a *variant* of TESLA in the QROM.
[b] In the benchmarks we include the improvements by Dagdelen et al. presented in [65].
[c] In the benchmarks we include the improvements by Ducas presented in [34].
[d] Benchmark on Haswell CPU from [16].
[e] The security of Rainbow5640 is based on the Multivariate Quadratic polynomial (MQ) and the Extended Isomorphism of Polynomials (EIP) problem, but no security reduction has been given yet.

## Acknowledgment

We thank Dennis Hofheinz and Vadim Lyubashevsky for initial discussions and Marc Fischlin and Florian Göpfert for fruitful comments.

## References

1. Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 572–590. Springer, April 2012. 3
2. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996. 18
3. Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *33rd ACM STOC*, pages 601–610. ACM Press, July 2001. 11
4. Martin Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. Algebraic algorithms for lwe problems. Cryptology ePrint Archive, Report 2014/1018, 2014. http://eprint.iacr.org/2014/1018/. 10
5. Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. On the complexity of the bkw algorithm on lwe. *Designs, Codes and Cryptography*, 74(2):325–354, 2015. 10
6. Martin R. Albrecht, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. Lazy modulus switching for the bkw algorithm on lwe. In Hugo Krawczyk, editor, *Public-Key Cryptography*, volume 8383 of *LNCS*, pages 429–445. Springer, 2014. 10
7. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, August 2009. 4
8. Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP 2011, Part I*, volume 6755 of *LNCS*, pages 403–415. Springer, July 2011. 10
9. László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986. 10
10. Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 28–47. Springer, February 2014. 2, 3, 5, 6, 8, 9, 10, 14
11. Rachid El Bansarkhani and Johannes Buchmann. Improvement and efficient implementation of a lattice-based signature scheme. In Tanja Lange, Kristin Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography*, volume 8282 of *LNCS*, pages 48–67. Springer, 2013. 14
12. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, November 1993. 20
13. Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 399–416. Springer, May 1996. 2, 20
14. Daniel J. Bernstein. A subfield-logarithm attack against ideal lattices. Post on the cr.yp.to blog, 2014. http://blog.cr.yp.to/20140213-ideal.html (accessed 2015-05-19). 2
15. Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Peter Schwabe, and Zooko Wilcox-O'Hearn. SPHINCS: practical stateless hash-based signatures. In Marc Fischlin and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 368–397. Springer, 2015. 2, 12, 14
16. Daniel J. Bernstein and Tanja Lange. eBACS: ECRYPT benchmarking of cryptographic systems. http://bench.cr.yp.to (accessed 2015-05-19). 14
17. DanielJ. Bernstein and Tanja Lange. Never trust a bunny. In Jaap-Henk Hoepman and Ingrid Verbauwhede, editors, *Radio Frequency Identification. Security and Privacy Issues*, volume 7739 of *LNCS*, pages 137–148. Springer, 2013. 10
18. Olivier Blazy, SaqibA. Kakvi, Eike Kiltz, and Jiaxin Pan. Tightly-secure signatures from chameleon hash functions. In Jonathan Katz, editor, *Public-Key Cryptography – PKC 2015*, volume 9020 of *LNCS*, pages 256–279. Springer, 2015. 23
19. Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. In *32nd ACM STOC*, pages 435–440. ACM Press, May 2000. 10

20. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, December 2011. 3, 5, 20, 21, 22
21. Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 361–379. Springer, August 2013. 21, 22
22. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013. 4
23. Peter Campbell, Michael Groves, and Dan Shepherd. Soliloquy: A cautionary tale. ETSI 2nd Quantum-Safe Crypto Workshop, 2014. http://docbox.etsi.org/Workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves_Annex.pdf. 2
24. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, May 2010. 23
25. Sanjit Chatterjee, Alfred Menezes, and Palash Sarkar. Another look at tightness. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 293–319. Springer, August 2011. 2
26. Anna Inn-Tung Chen, Ming-Shing Chen, Tien-Ren Chen, Chen-Mou Cheng, Jintai Ding, Eric Li-Hsiang Kuo, Frost Yu-Shuang Lee, and Bo-Yin Yang. SSE implementation of multivariate PKCs on modern x86 CPUs. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems – CHES 2009*, volume 5747 of *LNCS*, pages 33–48. Springer, 2009. 13, 14
27. Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer, December 2011. 10, 11
28. Yuanmi Chen and Phong Q. Nguyen. Bkz 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer, 2011. 11
29. Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 62–81. Springer, December 2013. 3, 5, 21
30. Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, volume 3531 of *LNCS*, pages 164–175. Springer, 2005. 13, 14
31. Alexendre Duc, Florian Tramèr, and Serge Vaudenay. Better algorithms for lwe and lwr. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 173–202. Springer, 2015. 10
32. Léo Ducas and Alain Durmus. Ring-LWE in polynomial rings. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 34–51. Springer, May 2012. 2
33. Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 40–56. Springer, August 2013. 2, 14
34. Léo Ducas. Accelerating bliss: the geometry of ternary polynomials. Cryptology ePrint Archive, Report 2014/874, 2014. http://eprint.iacr.org/2014/874/. 3, 14
35. MarcP.C. Fossorier, MiodragJ. Mihaljević, Hideki Imai, Yang Cui, and Kanta Matsuura. An algorithm for solving the lpn problem and its application to security evaluation of the hb protocols for rfid authentication. In Rana Barua and Tanja Lange, editors, *Progress in Cryptology - INDOCRYPT 2006*, volume 4329 of *LNCS*, pages 48–62. Springer, 2006. 10
36. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, May 2013. 2
37. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 3, 14
38. Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th ACM STOC*, pages 212–219. ACM Press, May 1996. 11
39. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547. Springer, September 2012. 2, 5, 14

40. Tim Güneysu, Tobias Oder, Thomas Pöppelmann, and Peter Schwabe. Software speed records for lattice-based signatures. In XXX, editor, *Post-Quantum Cryptography*, volume 7932 of *LNCS*, pages 67–82. Springer, 2013. 13, 14
41. Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In Joe P. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998. 2
42. Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 03*, pages 155–164. ACM Press, October 2003. 3, 6
43. Neal Koblitz and Alfred J. Menezes. Another look at "provable security". *Journal of Cryptology*, 20(1):3–37, January 2007. 20
44. Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS 2000*. The Internet Society, February 2000. 22
45. Po-Chun Kuo, Michael Schneider, Özgür Dagdelen, Jan Reichelt, Johannes Buchmann, Chen-Mou Cheng, and Bo-Yin Yang. Extreme enumeration on GPU and in clouds - - how many dollars you need to break SVP challenges -. In Bart Preneel and Tsuyoshi Takagi, editors, *CHES 2011*, volume 6917 of *LNCS*, pages 176–191. Springer, September / October 2011. 11, 12
46. Thijs Laarhoven and Benne de Weger. Faster sieving for shortest lattice vectors using spherical locality-sensitive hashing. In Kristin Lauter and Francisco Rodríguez-Henríquez, editors, *Progress in Cryptology – LATINCRYPT 2015*, LNCS. Springer, to appear. Preprint on `https://eprint.iacr.org/2015/211/`. 11, 12
47. Thijs Laarhoven, Michele Mosca, and Joop van de Pol. Finding shortest lattice vectors faster using quantum search. *Designs, Codes and Cryptography*, 2015. Online-first at `http://link.springer.com/article/10.1007/s10623-015-0067-5`. 11, 12
48. Éric Levieil and Pierre-Alain Fouque. An improved lpn algorithm. In Roberto De Prisco and Moti Yung, editors, *Security and Cryptography for Networks*, volume 4116 of *LNCS*, pages 348–359. Springer, 2006. 10
49. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, February 2011. 10
50. Mingjie Liu and PhongQ. Nguyen. Solving BDD by enumeration: An update. In Ed Dawson, editor, *Topics in Cryptology – CT-RSA 2013*, volume 7779 of *LNCS*, pages 293–309. Springer, 2013. 10
51. Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, December 2009. 3
52. Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, April 2012. 3, 5
53. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, May 2010. 2
54. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54. Springer, May 2013. 2
55. Daniele Micciancio. Improving lattice based cryptosystems using the Hermite normal form. In Joseph H. Silverman, editor, *Cryptography and Lattices*, volume 2146 of *LNCS*, pages 126–145. Springer, 2001. 4
56. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381. IEEE Computer Society Press, October 2004. 18
57. Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In Leonard J. Schulman, editor, *42nd ACM STOC*, pages 351–358. ACM Press, June 2010. 11
58. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009. 4
59. David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer, May 1996. 6
60. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. 4

61. Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.*, 66:181–199, 1994. 11
62. Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26:1484–1509, 1997. 1
63. Mark Zhandry. How to construct quantum random functions. In *53rd FOCS*, pages 679–687. IEEE Computer Society Press, October 2012. 22
64. Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, August 2012. 21
65. Özgür Dagdelen, Rachid El Bansarkhani, Florian Göpfert, Tim Güneysu, Tobias Oder, Thomas Pöppelmann, Ana Helena Sanchez, and Peter Schwabe. High-speed signatures from standard lattices. In Diego F. Aranha and Alfred Menezes, editors, *Progress in Cryptology – LATINCRYPT 2014*, volume 8895 of *LNCS*, pages 84–103. Springer, 2015. 2, 5, 9, 10, 12, 13, 14

## A Extended Definitions and Security Notions

### A.1 Smallest Integer Solution problem

The smallest integer solution (SIS) problem is formally defined (in the Euclidean norm) as follows.

**Definition 3 (Smallest Integer Solution Problem).** *Let $n, m, q > 0$ be integers and $\beta \in \mathbb{R}_{>0}$. The smallest integer solution problem $\mathsf{SIS}_{n,m,q,\beta}$ is $(t, \varepsilon)$-hard if for any algorithm $\mathcal{A}$, running in time $t$, we have*

$$\Pr\left[||\mathbf{x}|| \leq \beta \wedge (\mathbf{A}\mathbf{x} = \mathbf{0} \pmod q) \mid \mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}; \mathbf{x} \leftarrow \mathcal{A}(\mathbf{A})\right] \leq \varepsilon,$$

*where the probability is taken over the random choices of matrix $\mathbf{A}$ and algorithm $\mathcal{A}$.*

Similarly, the average-case instance of the SIS problem is hard as long as worst-case instances of GapSVP as shown in [2,56].

### A.2 Signature Schemes

A signature scheme $\Pi$ is defined as a tuple of the following algorithms: KeyGen, Sign, and Verify, where KeyGen and Sign are randomized algorithms.

Upon input the security parameter $\kappa$ the algorithm KeyGen outputs a publicly known verification key vk and a secret signing key sk. The algorithm Sign receives as input sk, vk, and a message $\mu$. It returns a valid signature $\sigma$ for the message $\mu$. The third algorithm Verify gets the verification key vk, the signature $\sigma$, and the message $\mu$ as input and checks if $\sigma$ is a valid signature for the message $\mu$. In this case Verify accepts and outputs 1, otherwise the algorithm outputs 0. We require as usual that the signature scheme has to be correct, meaning that for any message $\mu$, any $(\mathsf{sk}, \mathsf{vk}) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$ we have $\mathsf{Verify}(\mathsf{vk}, \mu, \mathsf{Sign}(\mathsf{sk}, \mu)) = 1$.

Let $\Pi = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a signature scheme and $\mathcal{A}$ be a probabilistic polynomial-time adversary which upon input the public verification key vk forges a signature. Then the scheme $\Pi$ is $(t_{\mathcal{A}}, q_h, q_s, \varepsilon_{\mathcal{A}})$-unforgeable if any algorithm $\mathcal{A}$ running in time $t_{\mathcal{A}}$, making at most $q_s$ queries to a singing oracle and $q_h$ hash queries to the random oracle outputs a forgery $(\mu^*, \sigma^*)$ such that $\mu^*$ was not queried to the signing oracle and $\mathsf{Verify}(\mathsf{vk}, \mu^*, \sigma^*) = 1$ is at most $\varepsilon_{\mathcal{A}}$.

### A.3 Lattices

A $k$-dimensional lattice $\Lambda$ is a discrete additive subgroup of $\mathbb{R}^n$ containing all integer linear combinations of $k$ linearly independent vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_k\}$ with $k \leq n$ and $n \geq 0$. More formally, we have $\Lambda = \{\ \mathbf{B} \cdot \mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^k\ \}$. The determinant of a lattice is the value $\det(\Lambda(\mathbf{B})) = \sqrt{\det\left(\mathbf{B}^\top \mathbf{B}\right)}$.

We note that a basis is not unique for a lattice and moreover, the determinant of a lattice is independent of the basis. That is, a different basis for the same lattice will yield the same determinant with the above formula.

Throughout this paper we are mostly concerned with $q$-ary lattices $\Lambda_q^{\perp}(\mathbf{A})$ and $\Lambda_q(\mathbf{A})$, where integer $q > 0$ denotes the modulus and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is a uniformly random chosen matrix. Lattices $\Lambda_q^{\perp}(\mathbf{A})$ and $\Lambda_q(\mathbf{A})$ are defined by

$$\Lambda_q^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^n \mid \mathbf{A}\mathbf{x} = \mathbf{0} \pmod q\},$$
$$\Lambda_q(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^n \mid \exists \mathbf{s} \in \mathbb{Z}^m \text{ s.t. } \mathbf{x} = \mathbf{A}^{\top}\mathbf{s} \pmod q\}.$$

Furthermore, for any $\mathbf{u} \in \mathbb{Z}_q^m$ we can define cosets $\Lambda_{\mathbf{u},q}^{\perp}(\mathbf{A}) = \{x \in \mathbb{Z}^n \mid \mathbf{A}\mathbf{x} = \mathbf{u} \pmod q\}$, i.e., $\Lambda_q^{\perp}(\mathbf{A}) = \Lambda_{\mathbf{0},q}^{\perp}(\mathbf{A})$. One can consider $\Lambda_{\mathbf{u},q}^{\perp}(\mathbf{A})$ as a *shifted lattice* by a vector $\mathbf{u}$, i.e., $\Lambda_{\mathbf{u},q}^{\perp}(\mathbf{A}) = \Lambda_q^{\perp}(\mathbf{A}) + \mathbf{y}$ where $\mathbf{y} \in \mathbb{Z}^m$ is an integer solution of $\mathbf{A}\mathbf{x} = \mathbf{u} \pmod q$. We note that for a uniformly chosen matrix $\mathbf{A}$ the determinant of the above modular lattices coincide with $\det(\Lambda_{\mathbf{u},q}^{\perp}(\mathbf{A})) = \det(\Lambda_q^{\perp}(\mathbf{A})) = q^{\operatorname{rank}(A)}$ for any $\mathbf{u} \in \mathbb{Z}_q^m$. Note that the rank of a uniformly random chosen matrix equals $\min(m, n)$ with high probability.

Given a measurable set $S$ and a lattice $L \subset \mathbb{Z}^n$, the *Gaussian heuristic* approximates the number of lattice points in $S$ by $|S \cap L| = \frac{vol(S)}{det(L)}$. Especially, if $L$ is a q-ary lattice $\Lambda_q^{\perp}(\mathbf{A})$ and $S = [-\delta, \delta]^n$, the *Gaussian heuristic* predicts the number of lattice points in $S$ by $|S \cap \Lambda_q^{\perp}(\mathbf{A})| = \frac{(2\cdot\delta+1)^n}{q^n}$.

## B Technical Proofs

*Proof (Lemma 2).* Recall that the rounding operator $\lfloor\cdot\rceil_d$ essentially drops component-wise the least $d$ bits of a vector. Thus, we can bound the probability in the theorem statement by the probability that there exists an $\mathbf{z} \in [-\delta, \delta]^n$ such that $\mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{y} + \mathbf{T}\mathbf{c} \pmod q$ multiplied by the factor of $2^{d\cdot n}$. This is because there are at most $2^{d\cdot n}$ possible values for $\mathbf{A}\mathbf{z}$ such that $\lfloor\mathbf{A}\mathbf{z}\rceil_d = \lfloor\mathbf{A}\mathbf{y} + \mathbf{T}\mathbf{c}\rceil_d \pmod q$. It remains to show that

$$\Pr[\,\exists\, \mathbf{z} \in [-\delta, \delta]^n \mid \mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{y} + \mathbf{T}\mathbf{c} \pmod q] \le \frac{(2\delta+1)^n}{q^m}. \tag{4}$$

Let $\mathbf{u} := \mathbf{A}\mathbf{y} + \mathbf{T}\mathbf{c} \pmod q$ and let $\mathbf{A}$ be of the form $\mathbf{A} = \begin{pmatrix}\mathbf{A_1}\\ \mathbf{A_2}\end{pmatrix}$ where $\mathbf{A_1} \in \mathbb{Z}_q^{n\times n}$ and $\mathbf{A_2} \in \mathbb{Z}_q^{m-n\times n}$, and $\mathbf{u} = \begin{pmatrix}\mathbf{u_1}\\ \mathbf{u_2}\end{pmatrix}$ where $\mathbf{u_1} \in \mathbb{Z}_q^n$ and $\mathbf{u_2} \in \mathbb{Z}_q^{m-n}$. Without loss of generality – by shifting the rows of $\mathbf{A}$ and $\mathbf{b}$ simultaneously – we can assume that $\mathbf{A_1}$ is a square matrix of $\operatorname{rank}(\mathbf{A_1}) = n$. This happens with high probability since $\mathbf{A}$ was sampled uniformly and $m > n$.

We stress that there exists only a single unique vector $\bar{\mathbf{z}} = (\bar{z}_1, \cdots, \bar{z}_n) \in \mathbb{Z}_q^n$ such that $\mathbf{A_1} \cdot (\bar{z}_1, \cdots, \bar{z}_n)^T = \mathbf{u_1} \pmod q$. We prove Equation (4), by pointing on two things: **(i)** the probability that $\bar{\mathbf{z}}$ is in $[-\delta, \delta]^n$ can be bounded by $(2\delta+1)^n/q^n$, and **(ii)** the probability that $\mathbf{A_2}\bar{\mathbf{z}} = \mathbf{u_2} \pmod q$ is bounded by $1/q^{m-n}$. Let us have a closer look at these two points.

**(i)** Define the set $S_{u_1} = \{\mathbf{z} \in [-\delta, \delta]^n \mid \mathbf{A_1}\mathbf{z} = \mathbf{u_1} \pmod q\}$. Since $\mathbf{A_1}$ is a random matrix of rank $n$, it defines a shifted random lattice $\Lambda_{u_1,q}^{\perp}(\mathbf{A_1}) = \{\mathbf{z} \in \mathbb{Z}^n \mid \mathbf{A_1}\mathbf{z} = \mathbf{u_1} \pmod q\}$ where $\det(\Lambda_{u,q}^{\perp}(\mathbf{A_1})) = q^n$. Via the *Gaussian heuristic* we approximate the number of lattice vectors in the set $S_{u_1}$ upper bounding the probability for **(i)**. We obtain

$$|[-\delta, \delta]^n \cap \Lambda_{u,q}^{\perp}(\mathbf{A_1})| = \frac{\operatorname{vol}([-\delta, \delta]^n)}{\det(\Lambda_{u,q}^{\perp}(\mathbf{A_1}))} = \frac{(2\delta+1)^n}{q^n}.$$

**(ii)** Let $a_{i,j}$ and $u_i$ denote the elements of $\mathbf{A}$ and $\mathbf{u}$, respectively. To bound the probability that $\mathbf{A_2}\bar{\mathbf{z}} = \mathbf{u_2} \pmod q$ holds, we basically bound the probability that $a_{i,1}\bar{z}_1 + \cdots + a_{i,n}\bar{z}_n = u_i$ holds for every $i = n+1, \cdots, m$. Note that this equation can be rewritten into the form

$u_i - a_{i,1}\bar{z}_1 - \ldots - a_{i,n-1}\bar{z}_{n-1} = a_{i,n}\bar{z}_n$. Let $\bar{z}_1, \ldots, \bar{z}_{n-1}$ be arbitrary. Then, the probability that $\bar{z}_n$ will satisfy the above equation is $1/q$ since the solution vector $\bar{\mathbf{z}}$ is distributed uniform in $\mathbb{Z}_q^n$ if $\mathbf{A_1}$ and $\mathbf{u_1}$ are. Since index $i$ ranges from $n+1$ to $m$, we get $1/q^{m-n}$ in total.

Taking the bounds given by **(i)** and **(ii)** together we show Equation (4), and obtain the theorem statement by multiplying with $2^{dn}$. □

*Proof (Lemma 3).* First, we bound the probability that there exist vectors $\mathbf{s}$ and $\mathbf{e}$ with small entries such that $\mathbf{As}+\mathbf{e}=\mathbf{t}$ for $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$ and $\mathbf{t} \xleftarrow{\$} \mathbb{Z}_q^m$, i.e., we show that

$$\delta := \Pr\left[\ \exists\ \mathbf{s} \in [-k\sigma, k\sigma]^n,\ \ \mathbf{e} \in [-k\sigma, k\sigma]^m \mid \mathbf{As}+\mathbf{e}=\mathbf{t}\ \right] \leq \frac{(2k\sigma+1)^{m+n}}{q^m}.$$

Since $\mathbf{A}$ and $\mathbf{t}$ are chosen uniformly random the probability $\delta$ can be bound by the ratio of the number of possible vectors $\mathbf{As}+\mathbf{e}$ with $\mathbf{s} \in [-k\sigma, k\sigma]^n$ and $\mathbf{e} \in [-k\sigma, k\sigma]^m$, and the number of possible vectors for $\mathbf{t} \xleftarrow{\$} \mathbb{Z}_q^m$, i.e.,

$$\delta \leq \frac{|\{\mathbf{As}+\mathbf{e} \mid \mathbf{s} \in [-k\sigma, k\sigma]^n \text{ and } \mathbf{e} \in [-k\sigma, k\sigma]^m\}|}{|\{\mathbf{t} \in \mathbb{Z}_q^m\}|}.$$

It holds that

$$\begin{aligned} |\{\mathbf{t} \in \mathbb{Z}_q^m\}| &= q^m, \\ |\{\mathbf{As}\ |\mathbf{s} \in [-k\sigma, k\sigma]^n\}| = |\{\mathbf{s} \in [-k\sigma, k\sigma]^n\}| &\leq (2k\sigma+1)^n \text{ since } \mathbf{A} \text{ is fixed, and} \\ |\{\mathbf{e} \in [-k\sigma, k\sigma]^m\}| &\leq (2k\sigma+1)^m. \end{aligned}$$

Thus,

$$\begin{aligned} |\{\mathbf{As}+\mathbf{e} \mid \mathbf{s} \in [-k\sigma, k\sigma]^n \text{ and } \mathbf{e} \in [-k\sigma, k\sigma]^m\}| &\leq (2k\sigma+1)^n \cdot (2k\sigma+1)^m \\ &= (2k\sigma+1)^{n+m}. \end{aligned}$$

Now, assume $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$ and $\mathbf{T} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$ are given. Let $\mathbf{s_i}$, $\mathbf{e_i}$, and $\mathbf{t_i}$ be the columns of $\mathbf{S}$, $\mathbf{E}$, and $\mathbf{T}$, respectively. Since the columns $\mathbf{s_i}$ and $\mathbf{s_j}$, and $\mathbf{e_i}$ and $\mathbf{e_j}$, are independent from each other for all $i, j \in \{1, \cdots, n\}$ , and $i \neq j$, it holds that

$$\begin{aligned} &\Pr\left[\ \exists\ \mathbf{S} \in [-k\sigma, k\sigma]^{n\times n} \text{ and } \mathbf{E} \in [-k\sigma, k\sigma]^{m\times n} \mid \mathbf{AS}+\mathbf{E}=\mathbf{T}\ \right] \\ &\leq \Pr\left[\ \forall i \in \{1, \cdots, n\}\ \exists\ \mathbf{s_i} \in [-k\sigma, k\sigma]^n,\ \ \mathbf{e_i} \in [-k\sigma, k\sigma]^m \mid \mathbf{As_i}+\mathbf{e_i}=\mathbf{t_i}\ \right] \\ &\leq \delta^n \\ &\leq \frac{(2k\sigma+1)^{m\cdot n+n^2}}{q^{m\cdot n}}\ . \end{aligned}$$

□

## C TESLA$_Q$: A Variant of TESLA Secure in the Quantum Random Oracle Model

We have proven the security of the signature scheme TESLA in the random oracle model (ROM) [12]. The random oracle is widely recognized as relevant for security reductions of cryptographic schemes and, in particular, extensively used for real-world cryptosystems, such as the signature schemes RSA-FDH [12, 13] and RSA-PSS [13]. Security in the ROM shows that in order to break the protocol, one must exploit some weaknesses of the hash function [43].

While security reductions in the ROM may be reasonable for classical adversaries, Boneh et al. [20] argue that this model potentially is inappropriate when considering quantum adversaries.

In fact, a quantum algorithm could implement the concrete choice of hash function — the instantiation of the random oracle — and run it in superposition on exponentially many inputs. To capture this capability of a quantum adversary, Boneh et al. introduced the quantum(-accessible) random oracle model (QROM). Here, adversaries are allowed to query the random oracle in superposition, simulating the real world capabilities of an adversary. Proofs in the QROM are significantly harder to obtain as common classical proof techniques do not easily transfer to the quantum setting, such as adaptive programmability, preimage awareness, lazy-sampling, and rewinding.

There are a couple of post-quantum schemes proven secure in the ROM which remain secure in the QROM [20,21,64]; however, it does not hold in general. In particular, rewinding the adversary is a delicate issue (such as in proofs of signatures derived via the Fiat-Shamir transformation [29]).

In this section, we show that a slight modification of TESLA facilitates the security reduction in the QROM. More precisely, we replace the input of the random oracle by its hash — for which we use a chameleon hash function. A chameleon hash function allows one to find efficiently collisions in its output if he is in possession of a trapdoor. In our security reduction the simulator makes use of such a trapdoor in order to simulate genuine signatures. In fact, by doing so, we are able to show that our security reduction of TESLA can be upgraded to be *history-free*; a sufficient property to claim security in the QROM even based on classical random oracles [20].

We call this modified construction $\mathsf{TESLA}_Q$ to emphasize that the construction provides security in the quantum random oracle model as opposed to TESLA. However, we do not rule out the possibility that TESLA could be proven secure against quantum adversaries in QROM, since, intuitively, this additional value is just introduced to overcome a technical dilemma and does not seem to add any security to the scheme. Moreover, we avoid proof techniques which are generally problematic in the quantum setting.

In the following we first recall some essential definitions and statements which will be of necessity to understand the security of $\mathsf{TESLA}_Q$ against quantum adversaries, and present its construction and corresponding security reduction afterwards.

## C.1 Preliminaries for the Quantum World

*Quantum Random-Oracle Model.* In the classical random oracle model a classical adversary has access to a random (hash) function $H : \{0,1\}^* \to \{0,1\}^\kappa$ which is model by a classical random oracle $O_c$. Thus, the adversary gets only the hash value at the classical state it asks for. In case the hash function is replaced by a concrete hash function instead of a random oracle, a quantum attacker can evaluate the concrete hash function on quantum states. To adjust the model to such an adversary Boneh et al. [20] introduce the *quantum(-accessible) random oracle model*. That means, the adversary is allowed to evaluate the random oracle in superposition, i.e., the adversary can submit quantum states $|\phi\rangle = \sum \alpha_x |x\rangle$ to the oracle $O_c$ and receives the hash value $\sum \alpha_x |O_c(x)\rangle$. Note that the quantum-accessible oracle can also be used as a classical oracle. Furthermore, honest parties and algorithms of the signature scheme are still classical and thus, access $O_c$ only via classical bit strings.

*History-Free Reduction.* The concept of history-free reduction, introduced by Boneh et al. [20], defines a subclass of security reductions for signature schemes in the classical random oracle model which imply security in the quantum model. Informally, history-freeness of a (classical) reduction proof means that the simulated response to a hash or sign query is independent of the (number of) responses to queries before, i.e., it is history-free. Unfortunately, only few signature schemes are known for which there are history-free reductions, and all those schemes follow the hash-and-sign paradigm. In contrast, our signature scheme $\mathsf{TESLA}_Q$ has a history-free reduction but follows the Fiat-Shamir transform – thus, does not require any trapdoor known to the signer. Unfortunately, chameleon hash functions involve trapdoors and will have a significant impact on the running time even though the trapdoor is merely used in our construction to simulate successfully signatures in our security reduction.

The following definition [20, Def. 4] captures history-free reductions formally.

**Definition 4 (History-free Reduction [20]).** *A signature scheme $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}^O, \mathsf{Verify}^O)$ in the classical random oracle model has a history-free reduction from a hard problem $\mathcal{P}$ if there is a proof of security that uses a classical PPT adversary $\mathcal{A}$ for $\mathcal{S}$ to construct a classical PPT algorithm $\mathcal{B}$ for problem $\mathcal{P}$ such that statements (a)-(e) are satisfied:*

*(a) Algorithm $\mathcal{B}$ for $\mathcal{P}$ consists of four explicit classical algorithms: $\underline{\mathsf{Start}}$, $\underline{\mathsf{Rand}}^{O_c}$, $\underline{\mathsf{Sign}}^{O_c}$, and $\underline{\mathsf{Finish}}^{O_c}$. The latter three algorithms have access to a shared classical random oracle $O_c$. These algorithms, except for $\underline{\mathsf{Rand}}^{O_c}$, may also make queries to the challenger for problem $\mathcal{P}$. The algorithms are used as follows:*
   *(i) Upon input an instance $x$ for problem $\mathcal{P}$, algorithm $\mathcal{B}$ first runs $\underline{\mathsf{Start}}(x)$ to obtain $(\mathsf{vk}, \mathsf{st})$ where $\mathsf{vk}$ is a signature verification key and $\mathsf{st}$ is private state to be used by $\mathcal{B}$. Algorithm $\mathcal{B}$ sends $\mathsf{vk}$ to $\mathcal{A}$ and plays the role of the challenger to $\mathcal{A}$.*
   *(ii) When $\mathcal{A}$ makes a classical random oracle query to $O(r)$, algorithm $\mathcal{B}$ responds with $\underline{\mathsf{Rand}}^{O_c}(r, \mathsf{st})$. Note that $\underline{\mathsf{Rand}}^{O_c}$ is given the current query as input, but is unaware of previous queries and responses.*
   *(iii) When $\mathcal{A}$ makes a classical signature query $\mathsf{Sign}(\mathsf{sk}, \mu)$, algorithm $\mathcal{B}$ responds with $\underline{\mathsf{Sign}}^{O_c}(\mu, \mathsf{st})$.*
   *(iv) When $\mathcal{A}$ outputs a signature forgery candidate $(\mu, \sigma)$, algorithm $\mathcal{B}$ outputs $\underline{\mathsf{Finish}}^{O_c}(\mu, \sigma, \mathsf{st})$.*

*(b) There is an efficiently computable function $\underline{\mathsf{Instance}}(\mathsf{vk})$ which produces an instance $x$ of problem $\mathcal{P}$ such that $\underline{\mathsf{Start}}(x) = (\mathsf{vk}, \mathsf{st})$ for some $\mathsf{st}$. Consider the process of first generating $(\mathsf{sk}, \mathsf{vk})$ from $\mathsf{KeyGen}(1^\kappa)$, and then computing $x = \underline{\mathsf{Instance}}(\mathsf{vk})$. The distribution of $x$ generated in this way is negligibly close to the distribution of $x$ generated as a real instance to $\mathcal{P}$.*

*(c) For fixed $\mathsf{st}$, consider the classical random oracle $O(r) = \underline{\mathsf{Rand}}^{O_c}(r, \mathsf{st})$. Define a quantum oracle $O_{quant}$, which transforms a basis element $|x, y\rangle$ into $|x, y \oplus O(x)\rangle$. We require that $O_{quant}$ is computationally indistinguishable from a random oracle for quantum algorithms.*

*(d) $\underline{\mathsf{Sign}}^{O_c}$ either aborts (and hence $\mathcal{B}$ aborts) or it generates a valid signature relative to the oracle $O(r) = \underline{\mathsf{Rand}}^{O_c}(r, \mathsf{st})$ with a distribution negligibly close to the correct signing algorithm. The probability that none of the signature queries abort is non-negligible.*

*(e) If $(\mu, \sigma)$ is a valid signature forgery relative to the public key $\mathsf{vk}$ and oracle $O(r) = \underline{\mathsf{Rand}}^{O_c}(r, \mathsf{st})$ then the output of $\mathcal{B}$ (i.e. $\underline{\mathsf{Finish}}^{O_c}(\mu, \sigma, \mathsf{st})$) causes the challenger for problem $\mathcal{P}$ to output 1 with non-negligible probability.*

Boneh et al. state that history-free reductions imply security in the quantum settings as in the following theorem:

**Theorem 2 ( [20]).** *Let $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a signature scheme. Suppose $\mathcal{S}$ has a history-free reduction from a problem $\mathcal{P}$. Further, assume that $\mathcal{P}$ is hard for polynomial-time quantum algorithms. Then, $\mathcal{S}$ is secure in the quantum-accessible random oracle model.*

Originally, Boneh et al. assumed further the existence of quantum-accessible pseudorandom functions. This assumption has been made obsolete by Zhandry [63] who presented secure constructions of such functions.

*Chameleon Hash Functions.* Chameleon hash functions has been introduced by Krawczyk and Rabin [44]. Roughly speaking, chameleon hash functions are like collision-resistant hash functions but come with a trapdoor that allows one to find collisions efficiently if in possession. The following captures the definition of chameleon hash functions formally.

**Definition 5 (Chameleon hash functions [21]).** *A chameleon hash function consists of three PPT algorithms $\mathcal{CH} = (\mathsf{Gen}, \mathsf{CH}, \mathsf{CH}^{-1})$ defined as follows. Upon input the security parameter $1^\kappa$, the probabilistic key generation algorithm $\mathsf{Gen}$ outputs a key pair $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{Gen}(1^\kappa)$ where $\mathsf{ek}$ is the evaluation key and $\mathsf{td}$ the corresponding trapdoor. The evaluation algorithm $\mathsf{CH}$ upon input an evaluation key $\mathsf{ek}$, a message $m \in \mathsf{M}_{\mathsf{ek}}$, and randomness $r \in \mathsf{R}_{\mathsf{ek}}$ outputs $\mathsf{CH}(\mathsf{ek}, m, r) \in \mathsf{Y}_{\mathsf{ek}}$. Chameleon hash functions satisfy the following properties:*

**Chameleon hash property** *The function $\mathsf{CH}^{-1}$ upon input trapdoor $\mathsf{td}$, messages $m, m' \in \mathsf{M}_{\mathsf{ek}}$, and randomness $r \in \mathsf{R}_{\mathsf{ek}}$ outputs $r' \leftarrow \mathsf{CH}^{-1}(\mathsf{td}, m, m', r)$ such that $\mathsf{CH}(\mathsf{ek}, m, r) = \mathsf{CH}(\mathsf{ek}, m', r')$.*

**Uniform distribution** *There exists a distribution $\mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$ such that for all $m \in \mathsf{M}_{\mathsf{ek}}$, the distributions* $(\mathsf{ek}, \mathsf{CH}(\mathsf{ek}, m, r)$ *and* $(\mathsf{ek}, y)$ *are computationally indistinguishable where* $(\mathsf{ek}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^\kappa)$, $r \xleftarrow{\$} \mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$ *and* $y \leftarrow \mathcal{U}(\mathsf{Y}_{\mathsf{ek}})$.

**Randomness indistinguishability** *Let $m, m' \in \mathsf{M}_{\mathsf{ek}}$ be arbitrary, and $r \xleftarrow{\$} \mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$. The distribution* $\mathsf{CH}^{-1}(\mathsf{td}, m, m', r)$ *is negligible close to the distribution $\mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$ where* $(\mathsf{ek}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^\kappa)$.

**Collision freeness** *For any PPT algorithm $\mathcal{A}$, we have*

$$\Pr[(m, r) \neq (m', r') \wedge \mathsf{CH}(\mathsf{ek}, m, r) = \mathsf{CH}(\mathsf{ek}, m', r') \mid (\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{Gen}(1^\kappa); (m, r, m', r') \xleftarrow{\$} \mathcal{A}(1^\kappa, \mathsf{ek})\ ] \leq \mathsf{negl}(\kappa)\ .$$

Several constructions of chameleon hash functions based on the lattice problems, such as the $\mathsf{SIS}$ assumption, are known [18, 24]. Any of those could be used as an instantiation in $\mathsf{TESLA}_Q$ as described in the next subsection.

## C.2 The Signature Scheme $\mathsf{TESLA}_Q$

Our signature scheme $\mathsf{TESLA}_Q$ is similar to $\mathsf{TESLA}$, but the value $\lfloor \mathbf{Ay} \rceil_d$ is first input to a chameleon hash function $\mathsf{CH}$ with fresh randomness $r \xleftarrow{\$} \mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$ before it is hashed (together with the message) by the random oracle. More formally, $\mathsf{TESLA}_Q = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ is defined as follows.

$\mathsf{KeyGen}$. The key generation is performs the same as steps as in the key generation algorithm of $\mathsf{TESLA}$ as described in Section 3.1.In addition, it generates a key pair of the chameleon hash function, $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{Gen}(1^\kappa)$ and sets $\mathsf{sk} = (\mathbf{S}, \mathbf{E})$ and $\mathsf{vk} = (\mathbf{T} = \mathbf{AS} + \mathbf{E}, \mathsf{ek})$.

$\mathsf{Sign}$. Upon input secret matrices $\mathbf{S}, \mathbf{E}$ and a message $\mu$ the signing algorithm performs the following. First, it samples vectors $\mathbf{y} \xleftarrow{\$} [-B, B]^n$ and $r \xleftarrow{\$} \mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$, and computes $\mathbf{v} \leftarrow \mathbf{Ay} \pmod q$. Afterwards, the hash value $c = H(\mathsf{CH}(\lfloor \mathbf{v} \rceil_d, r), \mu)$ is used to compute $\mathbf{z} \leftarrow \mathbf{y} + \mathbf{Sc}$ where $\mathbf{c} = F(c)$. Let $\mathbf{w} \leftarrow \mathbf{v} - \mathbf{Ec}$. If $|[\mathbf{w}_i]_{2^d}| > 2^{d-1} - L$ for some $i \in \{1, \cdots, m\}$ or $||\mathbf{z}||_\infty > B - U$, then the algorithm restarts; else, it outputs the signature $(\mathbf{z}, c, r)$.

$\mathsf{Verify}$. Upon input the public parameters, a message $\mu$ and signature $(\mathbf{z}, c, r)$, it first computes $\mathbf{c} = F(c)$ to obtain $\mathbf{w}' = \mathbf{Az} - \mathbf{Tc} \pmod q$, and returns 1 if $c = H(\mathsf{CH}(\lfloor \mathbf{w}' \rceil_d, r), \mu)$ and $||\mathbf{z}||_\infty \leq B - U$ are both satisfied; otherwise, it returns 0.

*Correctness.* It is easy to verify the correctness of $\mathsf{TESLA}_Q$. In the signature algorithm the value $\mathsf{CH}(\lfloor \mathbf{Ay} \rceil_d$,r) is hashed instead of $\lfloor \mathbf{Ay} \rceil_d$ as in $\mathsf{TESLA}$. Since the value $r \in \mathsf{R}_{\mathsf{ek}}$ is part of the signature, the verification algorithm is always able to verify a given valid signature.

## C.3 Security Reduction in the Quantum Setting

We show that by incorporating chameleon hash functions into $\mathsf{TESLA}$, we are able to give a history-free reduction from $\mathsf{LWE}$ to $\mathsf{TESLA}_Q$. Together with Theorem 2 we conclude that $\mathsf{TESLA}_Q$ is a quantum-secure lattice-based signature scheme in the quantum random oracle model. History-freeness is shown in the following lemma.

**Lemma 4.** *There exists a history-free reduction from $\mathsf{LWE}_{n,m,q,\sigma}$ to the signature scheme* $\mathsf{TESLA}_Q = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. *The tightness gap is equal to the signature scheme* $\mathsf{TESLA}$ *as stated in Theorem 1.*

*Proof.* Assume there is a quantum polynomial-time algorithm $\mathcal{A}$ which forges a valid signature in time $t_{\mathcal{A}}$ and probability $\varepsilon_{\mathcal{A}}$. We show how to construct a PPT algorithm $\mathcal{B}$ which solves the $\mathsf{LWE}_{n,m,q,\sigma}$ problem which internally makes black-box use of algorithm $\mathcal{A}$. That means, $\mathcal{B}$ upon an instance $(\mathbf{A}, \mathbf{T})$ decides whether matrices $\mathbf{A}$ and $\mathbf{T}$ are chosen uniformly random or whether they are of the form $\mathbf{AS} + \mathbf{E} = \mathbf{T}$ for some $\mathbf{S}$ and $\mathbf{E}$ with Gaussian-distributed entries. Following the idea of the classical reduction proof of Theorem 1, the adversary can forge (up to negligible probability) a signature only if a tuple $(\mathbf{A}, \mathbf{T})$ with $\mathbf{AS} + \mathbf{E} = \mathbf{T}$ is given. Furthermore, we show that our reduction is history-free.

A history-free reduction includes five (classical) algorithms $\underline{\mathsf{Start}}$, $\underline{\mathsf{Rand}}^{O_c}$, $\underline{\mathsf{Sign}}^{O_c}$, $\underline{\mathsf{Finish}}^{O_c}$, and $\underline{\mathsf{Instance}}$, as in Definition 4, where $\underline{\mathsf{Rand}}^{O_c}$, $\underline{\mathsf{Sign}}^{O_c}$, and $\underline{\mathsf{Finish}}^{O_c}$ have access to the same classical random oracle $O_c$.

**$\underline{\mathsf{Start}}$:** Algorithm $\mathcal{B}$ upon input $(\mathbf{A}, \mathbf{T})$ samples key pairs for the chameleon hash function $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{Gen}(1^\lambda)$ and defines $\mathsf{vk} = (\mathbf{T}, \mathsf{ek})$ and state $\mathsf{st} = (\mathsf{vk}, \mathsf{td})$. It returns $(\mathsf{vk}, \mathsf{st})$. Matrix $\mathbf{A}$ is set as the global system parameter.

**$\underline{\mathsf{Rand}}^{O_c}$:** When $\mathcal{A}$ queries $O(h, \mu)$ for some hash value $h \in \mathsf{Y}_{\mathsf{ek}}$, $\mathcal{B}$ responds with $\underline{\mathsf{Rand}}^{O_c}(h, \mu) := O_c(h, \mu)$.

**$\underline{\mathsf{Sign}}^{O_c}$:** When $\mathcal{A}$ asks the signature oracle on message $\mu$, algorithm $\mathcal{B}$ proceeds as follows. First, $\mathcal{B}$ samples values $\mathbf{v}' \xleftarrow{\$} \mathbb{Z}_q^m, \mathbf{r}' \xleftarrow{\$} \mathcal{D}_{\mathsf{R}_{\mathsf{ek}}}$ and $\mathbf{z} \xleftarrow{\$} [-B + U, B - U]^n$ uniformly at random. Afterwards, $\mathcal{B}$ queries its internal oracle $O_c$ on $(\mathsf{CH}(\lfloor \mathbf{v}' \rceil_d, r'), \mu)$ to obtain $c = O_c(\mathsf{CH}(\lfloor \mathbf{v}' \rceil_d, r'), \mu)$. Let $\mathbf{w} = \mathbf{Az} - \mathbf{Tc} \pmod q$ where $\mathbf{c} = F(c)$. Algorithm $\mathcal{B}$ responds with signature $\underline{\mathsf{Sign}}^{O_c}(\mu) := (\mathbf{z}, c, r)$ where $r = \mathsf{CH}^{-1}(\mathsf{td}, \lfloor \mathbf{v}' \rceil_d, \lfloor \mathbf{w} \rceil_d, r)$.

**$\underline{\mathsf{Finish}}^{O_c}$:** If $\mathcal{A}$ outputs a forgery $(\mu^*, \sigma^*) = (\mu^*, (\mathbf{z}^*, c^*, r^*))$ in time $t_\mathcal{A}$, then $\mathcal{B}$ outputs $\underline{\mathsf{Finish}}^{O_c}(\mu^*, \sigma^*) = 1$, i.e., $\mathcal{B}$ knows with overwhelming probability that $(\mathbf{A}, \mathbf{T})$ is an $\mathsf{LWE}$ tuple. Otherwise, $\underline{\mathsf{Finish}}^{O_c}$ outputs 0.

**$\underline{\mathsf{Instance}}$:** We define $\underline{\mathsf{Instance}}(\mathbf{A}, \mathbf{T}, \mathsf{ek}) := (\mathbf{A}, \mathbf{T})$. It holds that $\underline{\mathsf{Start}}(\underline{\mathsf{Instance}}(\mathbf{A}, \mathbf{T})) = ((\mathbf{A}, \mathbf{T}, \mathsf{ek}), (\mathbf{A}, \mathbf{T}, \mathsf{ek}, \mathsf{td}))$, thus $\underline{\mathsf{Instance}}$ and $\underline{\mathsf{Start}}$ satisfy the required property of Definition 4(b).

We now show that the remaining conditions for history-freeness are also satisfied.

Since $\underline{\mathsf{Rand}}^{O_c}$ returns a value given by the classical random oracle $O_c$, the output of $\underline{\mathsf{Rand}}^{O_c}$ is completely random and independent distributed, which shows property 4(c). The simulated responses to sign queries do not abort in any case. In fact, the distribution of the simulated signature are identical to genuine signatures by randomness indistinguishability of chameleon hash functions. Note that in the simulation we have $\mathsf{CH}(\lfloor \mathbf{v}' \rceil_d, r') = \mathsf{CH}(\lfloor \mathbf{Az} - \mathbf{Tc} \rceil_d, r)$ such that $c = O(\mathsf{CH}(\lfloor \mathbf{Az} - \mathbf{Tc} \rceil_d, r), \mu) = O(\mathsf{CH}(\lfloor \mathbf{v}' \rceil_d, r'), \mu) = O_c(\mathsf{CH}(\lfloor \mathbf{v}' \rceil_d, r'), \mu)$ holds for any message $\mu$. Thus, property 4(d) is satisfied.

Assume $\mathcal{A}$ outputs a valid signature forgery $(\sigma^*, \mu^*)$, where $\mathcal{A}$ did not ask $\mu^*$ to her signing oracle. Then, by the reduction proof of Theorem 1, we know that with overwhelming probability, the tuple $(\mathbf{A}, \mathbf{T})$ given as the challenge is an $\mathsf{LWE}$ tuple. If no signature is output by $\mathcal{A}$ in time $t_\mathcal{A}$, similarly with overwhelming probability it must have been two uniformly sampled matrices $(\mathbf{A}, \mathbf{T})$. Thus, $\mathcal{B}$ solves the $\mathsf{LWE}$ problem which shows 4(e). □

Using Theorem 2 and Lemma 4 we know that the signature scheme $\mathsf{TESLA}_Q$ is secure in the quantum-accessible random oracle model. We note that the security reduction for $\mathsf{TESLA}_Q$ in Theorem 3 is as tight as shown for $\mathsf{TESLA}$.

**Theorem 3.** *Let $\mathsf{TESLA}_Q = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be the signature scheme defined in C.2. If $\mathsf{LWE}_{n,m,q,\sigma}$ is hard against quantum adversaries, then the signature scheme $\mathsf{TESLA}_Q$ is unforgeable against adaptively chosen-message attacks in the quantum-accessible random oracle model.*