A preliminary version of this papers appears in the proceedings of PQCrypto 2017. This is the full version of the paper, updating the IACR ePrint Archive Record 2015/755.

Revisiting TESLA in the quantum random oracle model

Erdem Alkim[1], Nina Bindel[2], Johannes Buchmann[2], Özgür Dagdelen[3], Edward Eaton[4,5], Gus Gutoski[4], Juliane Krämer[2], and Filip Pawlega[4,5]

[1] Ege University, Turkey `erdemalkim@gmail.com`
[2] Technische Universität Darmstadt, Germany
`{nbindel, buchmann, jkraemer}@cdc.informatik.tu-darmstadt.de`
[3] BridgingIT GmbH, Germany `oezdagdelen@googlemail.com`
[4] ISARA Corporation, Canada
`{ted.eaton, gus.gutoski, filip.pawlega}@isara.com`
[5] University of Waterloo, Canada

May 4, 2017

Abstract. We study a scheme of Bai and Galbraith (CT-RSA'14), also known as TESLA. TESLA was thought to have a tight security reduction from the learning with errors problem (LWE) in the random oracle model (ROM). Moreover, a variant using chameleon hash functions was lifted to the quantum random oracle model (QROM). However, both reductions were later found to be flawed and hence it remained unresolved until now whether TESLA can be proven to be tightly secure in the (Q)ROM.
In the present paper we provide an entirely new, tight security reduction for TESLA from LWE in the QROM (and thus in the ROM). Our security reduction involves the adaptive re-programming of a quantum oracle. Furthermore, we propose parameter sets targeting 128 bits of security against both classical and quantum adversaries and compare TESLA's performance with state-of-the-art signature schemes.

Keywords: Quantum Random Oracle, Post Quantum Cryptography, Lattice-Based Cryptography, Signature Scheme, Tight Security Reduction

1 Introduction

Our interest in the present paper is in a quantum-resistant signature scheme proposed by Bai and Galbraith [8]. Those authors argue the security of their scheme via reductions from the *learning with errors (LWE)* and the *short integer solutions (SIS)* problems in the random oracle model (ROM). This scheme was subsequently studied by Alkim, Bindel, Buchmann, Dagdelen, and Schwabe under the name TESLA [4], who provided an alternate security reduction from the LWE problem only.

Since then, there have been several follow-up works on the Bai-Galbraith scheme [2, 4, 10, 61]. Most notably, a version of the scheme called ring-TESLA, whose security is based on the ring-LWE problem [2], has the potential to evolve into a practical, quantum-resistant signature scheme that might one day see widespread use as replacement for contemporary signature schemes such as ECDSA.

In what follows, we review the concepts of tightness and the quantum random oracle model as they relate to TESLA. We then list the contributions of the present paper and discuss related work by others.

1.1 Background

Security reduction and parameter choice. The security of digital signature schemes is often argued by reduction. A reductionist security argument typically proves a claim of the form, "any attacker $\mathcal{A}$ who can break the scheme can be used to build an algorithm $\mathcal{B}$ that solves some underlying hard computational problem". Hence, the security gap can be determined; it measures how much extra work $\mathcal{B}$ must perform in order to convert $\mathcal{A}$ into solving the underlying hard problem. If the run-time and probability of success of $\mathcal{B}$ are close to those of $\mathcal{A}$, *i.e.*, if the security gap is approximately 1, then the reduction is called *tight*. Achieving a small security gap, ideally a tight security reduction, is of theoretical interest in its own right, but it should also be an important consideration when selecting parameters for a concrete instantiation of a scheme. Specifically, the parameters of a signature scheme ought to be selected so that both (i) the effort needed to solve the underlying hard computational problem, and (ii) the security gap are taken into account. Hence, a tight security reduction is of advantage.

The need to instantiate schemes according to their security reductions and the role tight reductions play in these instantiations have been well argued by numerous authors. We refer the reader to [1, 21, 34] for a representative sample of these arguments.

The quantum random oracle model. Security arguments for the most efficient signature schemes—which therefore enjoy the most widespread real-world use—are typically presented in the ROM. (We refer to [37] by Koblitz and Menezes for discussion on why this might be the case.) The ROM postulates a truly random function that is accessible to attackers only through "black box" queries to an oracle for it—a random oracle. Any concrete proposal for a signature scheme must substitute a specific choice of hash function for the random oracle. An attacker armed with a quantum computer can be expected to evaluate that hash function in quantum superposition. Arguments that establish security even against such quantum-enabled attackers are said to hold in the quantum random oracle model (QROM).

It is conceivable that a signature scheme shown to be secure in the ROM may not be secure in the QROM. Thus, it is important that security arguments for quantum-resistant signature schemes hold not merely in the ROM, but also in the QROM.

Boneh *et al.* have proven that a security reduction in the ROM also holds in the QROM if it is *history-free* [16]. Unfortunately, many signature schemes have security reductions in the ROM that involve the *re-programming* of a random oracle; these reductions are not history-free. For these schemes, there remains a need to precisely clarify under what conditions these security reductions remain meaningful in the QROM.

Tightness in the QROM for TESLA. The security reduction presented by Bai and Galbraith for their signature scheme employs the Forking Lemma [53]. As such, it is non-tight and it involves re-programming, so it holds in the ROM but is not known to hold in the QROM.

As mentioned above, Alkim *et al.* presented an alternate security analysis for the Bai-Galbraith scheme, which they call TESLA. Their reduction is a tight reduction from LWE in the ROM. Moreover, those authors observed that their reduction can be made history-free at the cost of replacing a generic hash function with a chameleon hash function. It then follows from [16] that the history-free security reduction for TESLA holds also in the QROM. (Unfortunately, the use of a chameleon hash function would likely render any signature scheme too inefficient for widespread practical use.)

Unfortunately, a flaw in the original TESLA security reduction has been identified by the present authors. (The flaw was independently discovered by Chris Peikert.) This flaw is also present in several TESLA follow-up works, including ring-TESLA. As such, the status of the TESLA signature scheme and its derivative works has been open until now.

1.2 Our contribution

Our primary contributions are as follows:

New security reduction. We present a new security reduction from LWE to TESLA. Our new reduction is tight. It seems that the flaw in the original tight security reduction of TESLA does not admit a fix without a huge increase in the parameters; our new reduction is a significant re-work of the entire proof.

Security in the QROM with re-programming. Our new security reduction involves the adaptive re-programming of a random oracle and hence it is not history-free. Nevertheless, we show that it holds in the QROM by applying a seminal result from quantum query complexity due to Bennet, Bernstein, Brassard, and Vazirani [12]. It is possible that our approach can be abstracted so as to yield a general result on security reductions with re-programming in the QROM.

Our secondary contributions are as follows:

Parameter selection. We propose three sets of parameters for the concrete instantiation of TESLA: TESLA-0 and TESLA-1 targeting 96 and 128 bit security against a classical adversary, respectively; and TESLA-2, targeting 128 bits of security against a quantum adversary. All three parameter sets are chosen according to our (tight) security reduction.

The concrete parameter space admitted by our new security reduction is worse than that of previous reductions, but those previous reductions are either flawed or non-tight. Consequently, our proposed parameter sets lead to concrete instantiations of TESLA that are less efficient than previous proposals given in [4, 8, 61] that were not chosen according to the given security reduction.

Implementation. We provide a software implementation for the parameter sets TESLA-0 and TESLA-1. Our implementation targets Intel Haswell CPUs to provide a comparison of TESLA's performance with other signature schemes with different security levels. Unfortunately, the TESLA-2 parameter set does not seem to admit an implementation that can take advantage of the same fast parallel arithmetic instructions available on modern processors that were used in our implementations of TESLA-0 and TESLA-1, and so we do not provide a software implementation for TESLA at this parameter set. See Section 7 for details.

Gaussian Heuristic. All previous analyses of the scheme from [8] relied on an assumption known as the *"Gaussian heuristic"*, which provides an estimate for the number of lattice points contained within a bounding region. This estimate has been well studied, and is not accurate in general — particularly for small bounding bodies. However, its validity has been conjectured for simple bodies and lattices on which LWE and SIS are defined without any published proof. Although our approach does not rely on the Gaussian heuristic, we prove in Section 8 that its use in other works is in fact justified.

1.3 Related work

Tightness from "lossy" keys. In order to avoid the non-tightness inherent in the use of the Forking Lemma, we take an approach that was introduced by Katz and Wang to obtain tightly-secure signatures from the decisional Diffie-Hellman problem [34].

The idea is to use the underlying hardness assumption to show that "real", properly-formed public keys for the signature scheme are indistinguishable from "lossy", malformed public keys. The task of forging a signature for a lossy key is then somehow proven to be intractable.

Any attacker must therefore fail to forge when given a lossy public key. Thus, any attacker who succeeds in forging a signature when given a real public key can be used to distinguish real keys from lossy keys, contradicting the underlying hardness assumption.

In the case of TESLA, the real keys are matrices A and $\mathsf{T} = \mathsf{AS} + \mathsf{E}$ for some matrices S, E with small entries. (See Section 2.2 for a proper definition of these matrices and the LWE problem.) We call these real keys LWE yes-instances. The lossy keys are LWE no-instances: matrices A, T selected uniformly at random, so that the existence of S, E as above occurs with only negligible probability. We prove that the task of forging a TESLA signature for lossy keys is intractable, so that any TESLA forger must be able to solve the decisional LWE problem.

A Fiat-Shamir transform for "lossy" identification schemes. The TESLA signature scheme could be viewed as the result of applying the Fiat-Shamir transform to a "lossy" identification scheme based on LWE. A tight security reduction for TESLA then follows from a general theorem of Abdalla, Fouque, Lyubashevsky, and Tibouchi (AFLT theorem) on the tight security of any signature scheme obtained in this way [1].

In order to leverage the AFLT theorem, one must propose an identification scheme and prove that it is lossy. Such a proof could be obtained by excerpting the relevant parts of our security reduction to establish the simulatability and lossiness properties of a suitably chosen identification scheme. Such an exercise might make our rather monolithic security reduction easier to digest by modularizing it and phrasing it in a familiar framework.

However, security reductions obtained by applying the AFLT theorem are guaranteed to hold only in the ROM. In order to fully recover our security reduction from this framework, one must first re-prove the AFLT theorem in the QROM. This limitation is due to the fact that the proof of the AFLT theorem involves adaptively re-programming a hash oracle. As such, it does not meet any known conditions for lifting a given proof from the ROM into the QROM.

Given that our security reduction in the QROM also involves the adaptive re-programming of a hash oracle, perhaps our approach could be mined for insights to establish the AFLT theorem in the QROM.

Other tightly-secure LWE or SIS signature schemes. Gentry, Peikert, and Vaikuntanathan present a signature scheme with a tight security reduction from SIS in the ROM using a trapdoor construction based on possessing a secret short basis of a lattice [29]. Boneh *et al.* observed that the security reduction for this scheme is history-free, and thus holds in the QROM [16].

Boyen and Li present a signature scheme with a tight security reduction from SIS in the *standard model* [18], also using a short basis trapdoor. Since standard model security reductions do not rely on any assumptions about a random oracle, these reductions hold in the QROM.

The use of a short-basis trapdoor in a signature scheme imposes an additional constraint on the concrete parameter space admitted by that scheme's security reduction. This additional constraint on the parameters of short-basis trapdoor schemes seems to render them too inefficient for practical use. Since TESLA and its derivatives do not use a trapdoor construction, they do not suffer from this impediment.

Other than TESLA, we are aware of only one example of a signature scheme based on the Fiat-Shamir transform with a tight security reduction from LWE or SIS. Prior to Bai and Galbraith, a variant of a scheme by Lyubashevsky [42] was shown to admit a tight security reduction in the ROM by Abdalla *et al.* as part of an illustration of the aforementioned AFLT theorem [1]. An artifact of this reduction required Abdalla *et al.* to increase the parameters of the scheme, rendering it too inefficient for practical use. As mentioned earlier, security reductions produced via the AFLT theorem are not known to hold in the QROM.

Re-programming a quantum oracle. Adaptive reprogramming of a quantum oracle has been addressed in some specific cases. Unruh considered a reprogrammed quantum oracle in order to establish the security of a quantum position verification scheme [58]. It is not clear whether Unruh's results apply to our setting.

Eaton and Song present an asymptotic result on re-programming in the QROM [28] in a context quite different from ours. Since their result is asymptotic, it does not allow for concrete parameter selection, for which the tightness of the reduction needs to be explicit.

Our approach to re-programming is independent of these previous works, though some works—such as [16,28]—do draw upon the same result by Bennet *et al.* [12] that we employ. To our knowledge we are the first to present progress on re-programming in the QROM in the context of a cryptographic scheme with potential for quantum-resistant standardization.

A note on "lattice-based" cryptography. Part of the allure of cryptosystems based on LWE or SIS is that those problems enjoy worst-case to average-case reductions from fundamental problems about lattices such as the *approximate shortest independent vectors problem (SIVP)* or the *gap shortest vector problem (GapSVP)*. (See Regev [54] or the survey of Peikert [51] and the references therein.)

These reductions suggest that the ability to solve LWE or SIS on randomly chosen instances implies the ability to solve SIVP or GapSVP, even on the hardest instances. Indeed, cryptosystems based on LWE or SIS are often referred to as *lattice-based* cryptosystems, suggesting that the security of these cryptosystems ultimately rests upon the worst-case hardness of these lattice problems.

However, as observed by Chatterjee, Koblitz, Menezes, and Sarkar, existing worst-case to average-case reductions for LWE and SIS are highly non-tight [21]. We are not aware of a proposal for a concrete instantiation of a cryptosystem based on LWE or SIS with the property that the proposed parameters were selected according to such a reduction. Instead, it is common to instantiate such cryptosystems based on the best known algorithms for solving LWE or SIS. (In addition to TESLA, see for example [5, 17].)

For TESLA, we take care to instantiate the scheme according to its security reduction from LWE. However, we are unable to instantiate TESLA according to reductions from underlying lattice problems, due to the non-tightness of these reductions.

2 Preliminaries

In this section we clarify our notation used throughout the paper. Furthermore, we recall the learning with errors problem and give a short overview of used terms in quantum computing.

2.1 Notation

Integer scalars are denoted using Roman letters and if not stated otherwise, q is a prime integer in this paper. For any positive integer n the set $\mathbb{Z}_n$ of integers modulo n is represented by $\{-\lfloor (n-1)/2 \rfloor, \ldots, \lfloor n/2 \rfloor\}$. Fix a positive integer d and define the functions $[\cdot], [\cdot]_L : \mathbb{Z} \to \mathbb{Z}$ as follows. For any integer x let $[x]_L$ denote the representative of x in $\mathbb{Z}_{2^d}$, *i.e.*, $x = [x]_L \pmod{2^d}$, and let $[x] = (x - [x]_L)/2^d$. Informally, $[x]_L$ is viewed as the *least significant bits* of x and $[x]$ is viewed as the *most significant bits* of x. The definitions are easily extended to vectors by applying the operators for each component. An integer vector y is *B-short* if each entry is at most B in absolute value.

Vectors with entries in $\mathbb{Z}_q$ are viewed as column vectors and denoted with lowercase Roman letters in sans-serif font, *e.g.*, $\mathsf{y}, \mathsf{z}, \mathsf{w}$. Matrices with entries in $\mathbb{Z}_q$ are denoted with uppercase Roman letters in sans-serif font, *e.g.*, $\mathsf{A}, \mathsf{S}, \mathsf{E}$. The transpose of a vector or a matrix is denoted by v^T or M^T, respectively. We denote by $\|\mathsf{v}\|$ the Euclidean norm of a vector v, and by $\|\mathsf{v}\|_\infty$ its infinity norm. All logarithms are base 2. A function is called negligible in the security parameter λ, denoted by $negl(\lambda)$, if it decreases faster than the inverse of every polynomial in λ, for sufficiently large λ.

The centered discrete Gaussian distribution for $x \in \mathbb{Z}$ with standard deviation σ 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 $\rho_\sigma(\mathbb{Z}) = 1 + 2\sum_{x=1}^{\infty} \rho_\sigma(x)$.

For a finite set S, we denote sampling the element s uniformly from S with $s \leftarrow_\$ \mathcal{U}(S)$ or simply $s \leftarrow_\$ S$. Let χ be a distribution over $\mathbb{Z}$, then we write $x \leftarrow \chi$ if x is sampled according to χ. Moreover, we denote sampling each coordinate of a matrix $\mathsf{A} \in \mathbb{Z}^{m \times n}$ with distribution χ by $\mathsf{A} \leftarrow \chi^{m \times n}$ with $m, n \in \mathbb{Z}_{>0}$. 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. $\mathcal{A}^\chi$ denotes that $\mathcal{A}$ can request samples from the distribution χ.

2.2 The Learning with Errors Problem

In the following we recall the decisional learning with errors problem (LWE) and define its matrix varariant (M-LWE) [6].

Definition 1 (Learning with Errors Problem). *Let $n, m, q > 0$ be integers and χ be a distribution over $\mathbb{Z}$. For $\mathsf{s} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^n)$ define $\mathcal{A}_{\mathsf{s},\chi}$ to be the distribution*

[6] Note that, against common terminology in papers about module lattices, we do not mean *Module LWE* by the abbreviation M-LWE, but *Matrix LWE*.

that samples $\mathsf{a} \leftarrow_{\$} \mathbb{Z}_q^n$ *and* $\mathsf{e} \leftarrow \chi$ *and then returns* $(\mathsf{a}, \langle \mathsf{a}, \mathsf{s} \rangle + \mathsf{e}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. *The decisional learning with errors problem LWE*$_{n,m,q,\chi}$ *is* (t, ε)*-hard if for any algorithm* $\mathcal{D}$*, running in time* t *and making at most* m *queries to the distribution* $\mathcal{A}_{\mathsf{s},\chi}$*, it holds that*

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

We can also write m LWE instances to a secret $\mathsf{s} \in \mathbb{Z}_q^n$ as $(\mathsf{A}, \mathsf{A}\mathsf{s} + \mathsf{e} \pmod q)$ with $\mathsf{A} \leftarrow_{\$} \mathbb{Z}_q^{m \times n}$ and $\mathsf{e} \leftarrow \chi^m$.

The security of the signature scheme covered in this paper is based on the matrix version of LWE (M-LWE) defined in the following.

Definition 2 (Matrix Learning with Errors Problem). *Let* $n, n', m, q > 0$ *be integers and* χ *be a distribution over* $\mathbb{Z}$*. Define* $\mathcal{A}_{\mathsf{S},\chi}$ *to be the distribution that, for* $\mathsf{S} = (\mathsf{s}_1, ..., \mathsf{s}_{\mathsf{n}'})$ *with* $\mathsf{s}_1, ..., \mathsf{s}_{\mathsf{n}'} \leftarrow_{\$} \mathcal{U}\left(\mathbb{Z}_q^n\right)$*, samples* $\mathsf{a} \leftarrow_{\$} \mathbb{Z}_q^n$ *and* $e_1, ..., e_{n'} \leftarrow \chi$ *and then returns* $\left(\mathsf{a}^T, \mathsf{a}^T\mathsf{S} + (e_1, ..., e_{n'})\right) \in \mathbb{Z}_q^{1 \times n} \times \mathbb{Z}_q^{1 \times n'}$*. The matrix decisional learning with errors problem M-LWE*$_{n,n',m,q,\chi}$ *is* (t, ε)*-hard if for any algorithm* $\mathcal{D}$*, running in time* t *and making at most* m *queries to the distribution* $\mathcal{A}_{\mathsf{S},\chi}$*, it holds that*

$$\left| \Pr\left[\mathcal{D}^{\mathcal{A}_{\mathsf{S},\chi}}(\cdot) = 1\right] - \Pr\left[\mathcal{D}^{\mathcal{U}\left(\mathbb{Z}_q^n \times \mathbb{Z}_q^{n'}\right)}(\cdot) = 1\right] \right| \leq \varepsilon \; .$$

As before, m M-LWE samples to the secret matrix $\mathsf{S} = (\mathsf{s}_1, ..., \mathsf{s}_{n'}) \in \mathbb{Z}_q^{n \times n'}$ can be written as $(\mathsf{A}, \mathsf{A}\mathsf{S} + \mathsf{E}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times n'}$ with $\mathsf{A} \leftarrow_{\$} \mathbb{Z}_q^{m \times n}$ and $\mathsf{E} \leftarrow \chi^{m \times n'}$.

We call $(\mathsf{A}, \mathsf{T}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times n'}$ a **yes-instance** if there exists an $\mathsf{S} = (\mathsf{s}_1, ..., \mathsf{s}_{\mathsf{n}'})$ with $\mathsf{s}_1, ..., \mathsf{s}_{\mathsf{n}'} \in \mathbb{Z}_q^n$ and (A, T) are m M-LWE samples from the distribution $\mathcal{A}_{\mathsf{S},\chi}$. Otherwise, *i.e.*, when $(\mathsf{A}, \mathsf{T}) \leftarrow_{\$} \mathcal{U}\left(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times n'}\right)$, we call (A, T) a **no-instance**.

Theorem 1. *If LWE*$_{n,m,q,\chi}$ *is* $(\varepsilon/n', t)$*-hard then M-LWE*$_{n,n',m,q,\chi}$ *is* (ε, t)*-hard.*

Intuitively, the reduction loss exists since an adversary that can solve LWE has n' possibilities to solve M-LWE (see also [8, 17, 52]). The proof follows similar arguments as given in [52].

We note that the hardness of LWE (resp., M-LWE) is retained even if all coordinates of the secret vector s are sampled according to the error distribution χ, known as the "normal form" [6, 43]. We use the notation LWE$_{n,m,q,\sigma}$ if χ is distributed according to $\mathcal{D}_\sigma$. The LWE assumption comes with a worst-to-average-case reduction [20, 50, 54]; breaking certain average instances of LWE allows one to break all instances of certain standard lattice problems (namely GapSVP and SIVP).

2.3 Quantum Information and Quantum Oracles

We assume familiarity with the fundamentals of quantum information, such as the Dirac ket notation $|\cdot\rangle$ for pure quantum states and the density matrix formalism for mixed quantum states. (Recall that a mixed state can be viewed as a probabilistic mixture of pure states.) For background on quantum information the reader is referred to the books [35, 49].

The *trace norm* $\|\cdot\|_{\mathrm{Tr}}$ (also known as the *Schatten-1 norm* $\|\cdot\|_1$) is a matrix norm that generalizes the concept of statistical distance from probability distributions to quantum states. For example, the quantity $\|\rho - \rho'\|_{\mathrm{Tr}}$ captures the physically observable difference between two quantum states represented by density matrices ρ, ρ'.

A *channel* is a physically realizable map on mixed quantum states - the quantum analogue of a stochastic map applied to a classical probability distribution. Examples of channels include (i) *unitary channels*, which preserve purity of input states and act on classical basis states "in superposition", and (ii) *classical channels*, which measure input states in the standard basis and whose output is a probabilistic mixture of classical basis states. In this paper, hash oracles are unitary channels and sign oracles are classical channels.

Standard formalism for a quantum oracle for a hash function $\mathrm{H} : X \to Y$ specifies that the user has access to a unitary channel that applies the linear map $|x\rangle|y\rangle \mapsto |x\rangle|y + \mathrm{H}(x)\rangle$ on standard basis states, where $x \in X$ and $y \in Y$. It is understood that the range Y of $\mathrm{H}(\cdot)$ has a '+' operation defined on it with the property that $y + y' + y' = y$ for all $y, y' \in Y$, so that the unitary channel is its own inverse. Typically, one simply imagines that elements of y are written as binary strings, and the '+' operation is the bitwise exclusive-OR.

3 The Signature Scheme TESLA

In this section, we present the LWE-based signature scheme TESLA. Its orignal construction was proposed in 2014 by Bai and Galbraith [8]. It was later revisited by Dagdelen *et al.* [61] and by Alkim *et al.* [4].

TESLA's key generation, sign, and verify algorithms are listed informally in Algorithms 1, 2, and 3. More formal listings of these algorithms are given in Figure 1 in Section 5. Our proposed concrete parameter sets are derived in Section 5 and listed in Table 1.

Algorithm 1 KeyGen

```
Input: $\mathsf{A}$.
Output: Public key $\mathsf{T}$, secret key $(\mathsf{S}, \mathsf{E})$.
1: Choose entries of $\mathsf{S} \in \mathbb{Z}_q^{n \times n'}$ and $\mathsf{E} \in \mathbb{Z}_q^{m \times n'}$ from $\mathcal{D}_\sigma$.
2: If $\mathsf{E}$ has a row whose $h$ largest entries sum to $L$ or more then retry at step 1.
3: If $\mathsf{S}$ has a row whose $h$ largest entries sum to $L_S$ or more then retry at step 1.
4: $\mathsf{T} \leftarrow \mathsf{AS} + \mathsf{E}$.
5: Return public key $\mathsf{T}$ and secret key $(\mathsf{S}, \mathsf{E})$.
```

Parameters and Notation. TESLA is parameterized by positive integers q, m, n, n', h, d, B, L, L_S, U, a positive real σ, a hash oracle $\mathrm{H}(\cdot)$, and the publicly available matrix $\mathsf{A} \leftarrow_{\$} \mathbb{Z}_q^{m \times n}$. Let $\mathbb{H}$ denote the set of vectors $\mathsf{c} \in \{-1, 0, 1\}^{n'}$ with exactly h nonzero entries. For simplicity we assume that the hash oracle $\mathrm{H}(\cdot)$ has range $\mathbb{H}$, *i.e.*, we ignore the encoding function F, cf. Table 1. We call an integer vector w *well-rounded* if w is $(\lfloor q/2 \rfloor - L)$-short and $[\mathsf{w}]$ is $(2^d - L)$-short.

Deterministic signatures. We recommend that any implementation of TESLA employ standard techniques to achieve deterministic signatures. In particular, the key generation algorithm should produce a random seed as part of the secret key. The sign algorithm should use this seed, along with the input message μ, to derive a deterministic sequence of pseudorandom data that dictate the "random" choices of Algorithm 2. See Section 6 on implementation for further details.

Fixed-weight hash outputs. Any implementation of the hash oracle $\mathrm{H}(\cdot)$ will require an encoding function that embeds the output of a concrete hash function such as SHA-256 into the set $\mathbb{H}$ (see [31] for more information on embedding functions of this type). Naturally, the output length of the underlying hash function should be large enough so as to preclude collision attacks.

Additional checks in KeyGen and Sign. In contrast to earlier proposals [8, 61], we add two additional checks. The first one is the check in Line 3 in Algorithm 1. It ensures that no coefficient of the matrix S is too large, which allows for more concrete bounds during the security reduction. The parameter L_S is chosen such that the probability of rejecting S is smaller than $2^{-\lambda}$, cf. Section 5. The second additional check is in Line 5 in Algorithm 2. To ensure correctness of the scheme, it checks that the absolute value of each coordinate of $\mathsf{Ay} - \mathsf{Ec}$ is less or equal than $\lfloor q/2 \rfloor - L$.

Algorithm 2 Sign

Input: Message μ, secret key (S, E).
Output: Signature (z, c).

```
1: Choose $\mathsf{y}$ uniformly at random among $B$-short vectors from $\mathbb{Z}_q^n$.
2: $\mathsf{c} \leftarrow \mathrm{H}([\mathsf{Ay}], \mu)$.
3: $\mathsf{z} \leftarrow \mathsf{y} + \mathsf{Sc}$.
4: If $\mathsf{z}$ is not $(B - U)$-short then retry at step 1.
5: If $\mathsf{Ay} - \mathsf{Ec}$ is not well-rounded then retry at step 1.
6: Return signature $(\mathsf{z}, \mathsf{c})$.
```

Algorithm 3 Verify

Input: Message μ, public key (A, T), purported signature (z, c).
Output: "Accept" or "reject".

```
1: If $\mathsf{z}$ is not $(B - U)$-short then reject.
2: If $\mathrm{H}([\mathsf{Az} - \mathsf{Tc}], \mu) \neq \mathsf{c}$ then reject.
3: Accept.
```

4 Security Reduction for TESLA

Our main theorem on the security of TESLA informally states that as long as M-LWE can not be solved in time t and with success probability ε then no adversary $\mathcal{A}$ exists that can forge signatures of TESLA in time t' and with success probability ε', if $\mathcal{A}$ is allowed to make at most q_h hash und q_s sign queries. The main theorem is as follows.

Theorem 2 (Security of TESLA). *Let q, m, n, n', h, d, B, L, L_S, U, σ, λ, κ be TESLA parameters that are convenient*[7] *(according to Definition 3 in Appendix D) and that satisfy the bounds in Table 1.*

If M-LWE is (t, ε)-hard then TESLA is existentially $(t', \varepsilon', q_h, q_s)$-unforgeable against adaptively chosen message attacks with $t' \approx t$ in (i) the quantum *random oracle model with*

$$\varepsilon' < \varepsilon + \frac{3}{2^\lambda} + \frac{2^{m(d+1)+3\lambda+1}}{q^m}(q_h + q_s)^2 q_s^3 + 2(q_h + 1)\sqrt{\frac{1}{2^h \binom{n'}{h}}}, \tag{1}$$

and in (ii) the classical *random oracle model with*

$$\varepsilon' < \varepsilon + \frac{3}{2^\lambda} + \frac{2^{m(d+1)+3\lambda+1}}{q^m}(q_h + q_s)^2 q_s^3 + q_h \frac{1}{2^h \binom{n'}{h}}. \tag{2}$$

The proof of Theorem 2 is given in appendices A, B, C, D. Here we present a sketch of this proof and a selection of some intermediate results we feel are the most significant technical contributions of the present manuscript.

Let $\mathcal{F}$ be a forger that forges signatures of the TESLA scheme with probability $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$, where $\text{forge}(\mathsf{A}, \mathsf{T})$ denotes the event that $\mathcal{F}$ forges a signature on input (A, T), which is a yes- or a no-instance of LWE. We build an LWE-solver $\mathcal{S}$ whose run time is close to that of $\mathcal{F}$ and who solves LWE with success bias close to $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$. It then follows from the presumed hardness of LWE that $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$ must be small.

Given an LWE input (A, T), the LWE-solver $\mathcal{S}$ treats (A, T) as a TESLA public key; $\mathcal{S}$ runs $\mathcal{F}$ on input (A, T) and outputs "yes" if and only if $\mathcal{F}$ succeeds in forging a TESLA signature.

[7] It is not necessary that TESLA parameters be convenient in order to derive negligibly small upper bounds on ε'; the definition of convenience merely facilitates a simplified statement of those bounds.

In order to run $\mathcal{F}$, the LWE-solver $\mathcal{S}$ must respond in some way to $\mathcal{F}$'s quantum queries to the hash oracle and to $\mathcal{F}$'s classical queries to the sign oracle. Our description of $\mathcal{S}$ includes a procedure for responding to these queries.

That $\mathcal{S}$ solves LWE with success bias close to $\Pr[\text{forge}(\mathsf{A},\mathsf{T})]$ is a consequence of the following facts:

1. For yes-instances of LWE, the probability with which $\mathcal{S}$ outputs "yes" is close to $\Pr[\text{forge}(\mathsf{A},\mathsf{T})]$.
2. For no-instances of LWE, $\mathcal{F}$ successfully forges (and hence $\mathcal{S}$ outputs "yes") with only negligible probability.

4.1 Yes-Instances of LWE

We argue that $\mathcal{S}$'s responses to $\mathcal{F}$'s oracle queries are indistinguishable from the responses $\mathcal{F}$ would receive from real oracles, from which it follows that $\mathcal{S}$ reports "yes" with probability close to $\Pr[\text{forge}(\mathsf{A},\mathsf{T})]$.

Each time $\mathcal{S}$ simulates a call to the sign oracle, it must "re-program" its simulated hash oracle on one input. Because $\mathcal{F}$ is permitted to make quantum queries to the hash oracle, we must show that $\mathcal{F}$ is unlikely to notice when a quantum random oracle has been re-programmed.

To this end, let $\mathbb{Y}$ denote the set of vectors $\mathsf{y} \in \mathbb{Z}_q^n$ such that y is B-short and define the following quantities for each choice of TESLA keys $(\mathsf{A},\mathsf{T}),(\mathsf{S},\mathsf{E})$:

nwr(A,E): The probability over $(\mathsf{y},\mathsf{c}) \in \mathbb{Y}\times\mathbb{H}$ that $\mathsf{Ay}-\mathsf{Ec}$ is not well-rounded.
coll(A,E): The maximum over all $\mathsf{w} \in \{[\mathsf{x}] : \mathsf{x} \in \mathbb{Z}_q^m\}$ of the probability over $(\mathsf{y},\mathsf{c}) \in \mathbb{Y}\times\mathbb{H}$ that $[\mathsf{Ay}-\mathsf{Ec}] = \mathsf{w}$.

We prove the following in Appendix B.6, cf. Proposition 7.

Proposition 1 (Re-Programming in TESLA, Informal Statement). *The following holds for each choice of TESLA keys* $(\mathsf{A},\mathsf{T}),(\mathsf{S},\mathsf{E})$*, each hash oracle* $\mathrm{H}(\cdot)$*, and each* $\gamma > 0$.

Suppose the quantum state ρ_{H} *was prepared by some party* $\mathcal{D}$ *using* t *quantum queries to* $\mathrm{H}(\cdot)$. *Let* $\mathrm{H}'(\cdot)$ *be a hash oracle that agrees with* $\mathrm{H}(\cdot)$ *except on a small number of randomly chosen inputs* $(\cdot,\mu)$ *for each possible message* μ. *Let* $\rho_{\mathrm{H}'}$ *be the state prepared when* $\mathcal{D}$ *uses hash oracle* $\mathrm{H}'(\cdot)$ *instead of* $\mathrm{H}(\cdot)$.

Then $\|\rho_{\mathrm{H}'}-\rho_{\mathrm{H}}\|_{\mathrm{Tr}} < \gamma$ *except with probability at most*

$$\frac{t^2}{\gamma^2}\cdot\frac{\mathrm{coll}(\mathsf{A},\mathsf{E})}{1-\mathrm{nwr}(\mathsf{A},\mathsf{E})} \tag{3}$$

over the choice of inputs upon which $\mathrm{H}(\cdot)$ *and* $\mathrm{H}'(\cdot)$ *differ.*

We also prove bounds on nwr(A,E) and coll(A,E) that hold with high probability over the choice of TESLA keys $(\mathsf{A},\mathsf{T}),(\mathsf{S},\mathsf{E})$.

4.2 No-Instances of LWE

We argue that, except with negligibly small probability over the choice of hash oracle $\mathrm{H}(\cdot)$ and LWE no-instance (A,T), a TESLA forger cannot forge a signature for (A,T) without making an intractably large number of queries to the hash oracle.

To forge a signature for message μ, a forger must find a hash input (w,μ) whose output $\mathsf{c} = \mathrm{H}(\mathsf{w},\mu)$ has the property that there exists a $(B-U)$-short $\mathsf{z} \in \mathbb{Z}_q^n$ for which $[\mathsf{Az} - \mathsf{Tc}] = \mathsf{w}$. Let $\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T}) \subset \mathbb{H}$ denote the set of all such c. A hash input (w,μ) is called *good* for $\mathrm{H}(\cdot)$ and (A,T) if $\mathrm{H}(\mathsf{w},\mu) \in \mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})$. (Once a good hash input has been found, the forger must then somehow *find* the vector z witnessing this fact. For our purpose, we assume that the forger gets it for free.)

For each LWE no-instance (A,T), a given hash input (w,μ) is good for $\mathrm{H}(\cdot)$ and (A,T) with probability

$$\frac{\#\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})}{\#\mathbb{H}} \tag{4}$$

over the choice of hash oracle $\mathrm{H}(\cdot)$. In Appendix C we argue that, except with negligibly small probability over the choice of $\mathrm{H}(\cdot)$ and (A,T), the fraction of hash inputs that are good is at most the expectation over LWE no-instances (A,T) of the ratio (4), maximized over all $\mathsf{w} \in \{[\mathsf{x}] : \mathsf{x} \in \mathbb{Z}_q^m\}$. We then prove the following (*cf.* Proposition 9 in Appendix C) .

Proposition 2 (Good Hash Inputs are Rare). *If the TESLA parameters are convenient (according to Definition 3 in Appendix D) then*

$$\mathop{\mathrm{Ex}}_{(\mathsf{A},\mathsf{T})}\left[\max_{\mathsf{w}}\left\{\frac{\#\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})}{\#\mathbb{H}}\right\}\right] \le \frac{1}{\#\mathbb{H}}. \tag{5}$$

Thus, the fraction of good hash inputs is at most $1/\#\mathbb{H}$ except with vanishingly small probability over the choice of hash oracle $\mathrm{H}(\cdot)$ and LWE no-instance (A,T).

Since each hash input is good with a fixed probability independent of other hash inputs, the only way to discover a good input is via search through an unstructured space. It then follows from known lower bounds for quantum search over an unstructured space that the forger cannot find a good hash input—and thus a TESLA forgery—using only q_h quantum queries to the hash oracle.

5 Selecting Parameters for TESLA

In this section we propose parameter sets for TESLA. Table 1 illustrates our concrete choice of parameters and Table 2 gives the hardness of the corresponding LWE instances. We propose three parameter sets: TESLA-0 that targets the same (classical) bit security of 96 bit as the instantiation proposed in [61], called DEG^+. TESLA-1 targets 128 bit of classical security and TESLA-2 targets 128 bit of security against quantum adversaries. Note that the parameter set DEG^+ was orignally proposed to give 128 bit of security, *i.e.*, $\lambda = 128$, but due to new methods to estimate the bit security its bit security is now only 96 bit.

Table 1. Concrete instantiation TESLA-2 of 128 bit of security against classical and quantum adversaries, and TESLA-0 of 96 bit and TESLA-1 of 128 bit of security against classical adversaries; comparison with the instantiation proposed in [61], called DEG$^+$, of 96 bit security (classically); sizes are given in kilo byte [KB]; sizes are theoretic sizes for fully compressed keys and signatures; for sizes used by our software see Table 3.

Parameter	Bound	DEG$^+$	TESLA-0	TESLA-1	TESLA-2
λ		128	96	128	128
κ		256	256	256	256
n		532	644	804	1300
n'		532	390	600	1036
m		840	3156	4972	4788
σ	$> 2\sqrt{n}$	43	55	57	73
L	$3h\sigma$ or $2.8h\sigma$, see Sec. 5.1	2322	5082	6703	17987
L_S	$14\sigma h$	-	25410	33516	89936
h	$2^h\binom{n'}{h} \geq 2^{3\lambda}$ (classically)	18	33	42	-
	$2^h\binom{n'}{h} \geq 2^{5\lambda}$ (quantumly)	-	-	-	88
B	$\geq 14n\sqrt{h}\sigma$	$2^{21}-1$	$2^{22}-1$	$2^{22}-1$	$2^{24}-1$
U	$\lceil 14\sqrt{h}\sigma \rceil$	2554	4424	5172	9588
d	$(1-2L/2^d)^m \geq 0.3$	23	25	26	27
q	satisfying the bound in Eq. 152, $\geq \left(2^{m(d+1)+4\lambda+1}(q_h+q_s)^2 q_s^3\right)^{1/m}$	$2^{29}-3$	$2^{31}-99$	$2^{31}-19$	40582171961 $\approx 2^{35.24}$
δ_{KeyGen}	empirically, see Sec. 5.1	0.99	1	1	future work
δ_{Sign}		0.314	0.307	0.154	future work
H	$\{0,1\}^* \to \{0,1\}^\kappa$	SHA-256			
F	$\{0,1\}^\kappa \to \mathbb{H}_{n',\omega}$	see [31]			
PRF_1	$\{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^\kappa$	-	SHA-256		
PRF_2	$\{0,1\}^\kappa \times \mathbb{Z} \to [-B,B]^n$	-	ChaCha20		
public-key size	$mn'\lceil \log_2(q) \rceil$	1 582	4 657	11 288	21 799
secret-key size	$(nn'+mn')\lceil \log_2(14\sigma) \rceil$	891	1 809	4 230	7 700
signature size	$n\lceil \log_2(2(B-U)) \rceil + \kappa$	1.4	1.8	2.3	4.0

5.1 Derivation of System Parameters

Our security reduction for TESLA minimizes the underlying assumptions which allows us to choose secure parameters from a greater set of choices compared to [8,61]. More precisely, our parameters do not have to involve a hard instance of the SIS assumption as it was done by Bai and Galbraith [8] before. We summarize the bounds and conditions of each parameter in Table 1 and explicate the computation of some of the listed parameters in the following. Furthermore, we state the resulting key and signature sizes in the table.

Compared to [8,61], we introduce the parameter n' as the column dimension of the secret matrices S and E to get more flexibility in the choice of parameters. The value of n' influences the parameters h (and hence B, U, q, and the encoding function F) and the size of the secret key.

Another important parameter of the signature scheme is the value L. In the original work [8], it is set to $L = 7h\sigma$, whereas it is set to $L = 3h\sigma$ in [61]. We choose L to be roughly $L = 2.8h\sigma$. We note that the smaller the value L, the higher the probability of acceptance in the signature algorithm (Line 9, Figure 1) becomes.

We add checkS to the key generation algorithm and the corresponding parameter L_S to bound $\|\mathsf{Sc}\| \leq L_S$ in the security reduction. We determine the value L_S such that S is rejected only with negligibly small (in the security parameter λ) probability. Hence, we do not decrease the size of the key space further. We choose L_S to be $14h\sigma$.

The acceptance probabilities of a signature δ_{Sign} and of a secret key (S, E) in Table 1 are determined experimentally.

To ensure both correctness and security of our signature scheme, we choose parameters with respect to our reduction, hence, we choose parameters such that $\epsilon' \approx \epsilon$ in Equation (1) and (2). We propose to choose $q_h \leq 2^{\lambda}$ and $q_s \leq 2^{\lambda/2}$, since a hash query is merely the evaluation of a publicly available function and hence the adversary can use all its computational power to make hash queries. The number of sign queries is somewhat limited since it involves more complicated operations. We refer to [36] (especially, Section 7) for further discussion.

5.2 Concrete Bit Security of TESLA

The security reduction given in Section 4 provides a reduction from the hardness of M-LWE and bounds *explicitly* the forging probability with the success probability of the reduction. More formally, let $\varepsilon_{\mathcal{A}}$ and $t_{\mathcal{A}}$ denote the success probability and the run time of a forger $\mathcal{A}$ against our signature scheme and let $\varepsilon_{\mathcal{B}}$ and $t_{\mathcal{B}}$ denote analogous quantities for the reduction $\mathcal{B}$ presented in the proof of Theorem 2. We say that M-LWE is *η-bit hard* if $t_{\mathcal{B}}/\varepsilon_{\mathcal{B}} \geq 2^{\eta}$; and we say that the signature scheme is *λ-bit secure* if $t_{\mathcal{A}}/\varepsilon_{\mathcal{A}} \geq 2^{\lambda}$.

Since we choose parameters such that $\varepsilon_{\mathcal{B}} \approx \varepsilon_{\mathcal{A}}$ and $t_{\mathcal{B}} \approx t_{\mathcal{A}}$, the bit hardness of the M-LWE instance is about the same as the bit security of our signature scheme instantiated as described below.

However, we lose $\lceil \log(n') \rceil$ bits of security due to the reduction from LWE to M-LWE. Hence, we have to choose LWE instances of $\lambda + \lceil \log(n') \rceil$ bit hardness. We explain how we choose those LWE instances in the following.

Estimation of the Bit Hardness of LWE Against Classical Adversaries. Albrecht, Player, and Scott [3] presented the *LWE-Estimator*, a software to estimate the *classical* hardness of LWE given the matrix dimension n, the modulus q, and the relative error rate $\alpha = \frac{\sqrt{2\pi}\sigma}{q}$. The original LWE-Estimator does not take the number of given LWE samples into account, leading to a more conservative choice of parameters for applications where only a restricted number of LWE samples is given. Recently, a fork of the original LWE-Estimator was presented [56, 57] that takes a given number of samples m into consideration. Since for TESLA only a restricted number of m LWE samples is given, we use [57] to estimate the classical bit hardness of our proposed LWE instances.

We summarize the classical estimations of the most efficient LWE solvers for our instances in Table 2, namely, the standard embedding or approach by Kannan [33], the dual embedding [8], and the decoding attack [7,40,41]. We also update the estimation of the bit hardness of the LWE instances proposed in [61].

Table 2. Estimation of the hardness of LWE instances given in TESLA-0, TESLA-1, and TESLA-2 against the decoding attack and the (dual and standard) embedding approach, in comparison to the parameter sets proposed by Dagdelen et al. [61], called DEG$^+$; estimations are computed using the LWE-Estimator with a restricted number of samples [3,57].

Attack	DEG$^+$	TESLA-0	TESLA-1	TESLA-2
Classical Hardness [bit]				
Decoding	156	**110**	**142**	**204**
Dual Embedding	**96**	**110**	**142**	205
Standard Embedding	164	111	143	205
Post-Quantum Hardness [bit]				
Decoding	73	74	98	146
Dual Embedding	**61**	**71**	**94**	**142**
Standard Embedding	111	**71**	95	**142**

Estimation of the Bit Hardness of LWE Against Quantum Adversaries. Recently the first proposals on how to estimate the hardness of LWE against quantum adversaries were published [5, 10, 17]. However, those do not follow the same methodology as the LWE-Estimator we used to estimate the classical hardness of LWE. Although the LWE-Estimator originally considers only classical hardness estimations, the first changes towards quantum estimations were made. In the latest version (from commit-id b929691 on) the estimates for a quantumly enhanced sieving algorithm [38] are included as a subroutine to solve the lattice reduction algorithm BKZ2.0 [23].

As mentioned in [5], a recently published quantum algorithm [47] can be applied to enumeration methods, possibly giving a quadratic speed-up on the run time of enumeration. We apply the quadratic speed-up to the currently fastest enumeration estimations by Micciancio and Walter [46] and add the resulting estimations as a subroutine to be used in BKZ2.0. We summarize the estimations using quantum sieving and quantum enumeration in Table 2.

6 Software Implementation

To evaluate the performance of our proposed parameter sets we present a software implementation targeting the Intel Haswell microarchitecture. The starting point for our implementation is the software presented by Dagdelen et al. [61], which we obtained from the authors. Our software offers the same level of protection against timing attacks as the software presented in [61]. The software makes use of the fast AVX2 instructions on vectors of 4 double-precision floating-point numbers. In this section we first give a description of TESLA with technical details necessary for implementations. Furthermore, we present important modifications to the software proposed in [61].

```
Algorithm KeyGen

INPUT: $1^\lambda; \mathsf{A}, n, n', m, q, \sigma$
OUTPUT: $(\mathsf{S}, \mathsf{E}, s), \mathsf{T}$

1. $\mathsf{S} \leftarrow_\$ \mathcal{D}_\sigma^{n \times n'}$
2. $\mathsf{E} \leftarrow_\$ \mathcal{D}_\sigma^{m \times n'}$
3. if $\mathsf{checkE}(\mathsf{E}) = 0 \vee \mathsf{checkS}(\mathsf{S}) = 0$
4.     then Restart
5. $s \leftarrow_\$ \{0,1\}^\kappa$
6. $\mathsf{T} \leftarrow \mathsf{AS} + \mathsf{E} \pmod q$
7. $\mathrm{sk} \leftarrow (\mathsf{S}, \mathsf{E}, s), \mathrm{pk} \leftarrow \mathsf{T}$
8. return $(\mathrm{sk}, \mathrm{pk})$
```

```
Algorithm Verify

INPUT: $\mu, q, \mathsf{z}, c, \mathsf{A}, \mathsf{T}$
OUTPUT: {0,1}

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

```
Algorithm Sign

INPUT: $\mu, q, \mathsf{A}, \mathsf{S}, \mathsf{E}, s$
OUTPUT: $(\mathsf{z}, c)$

1. $j \leftarrow 0$
2. $\mathsf{k} \leftarrow PRF_1(s, \mu)$
3. $\mathsf{y} \leftarrow PRF_2(\mathsf{k}, j)$
4. $\mathsf{v} \leftarrow \mathsf{Ay} \pmod q$
5. $c \leftarrow H(\lfloor \mathsf{v} \rceil, \mu)$
6. $\mathsf{c} \leftarrow F(c)$
7. $\mathsf{z} \leftarrow \mathsf{y} + \mathsf{Sc}$
8. $\mathsf{w} \leftarrow \mathsf{v} - \mathsf{Ec} \pmod q$
9. if $\| [\mathsf{w}]_L \|_\infty > 2^{d-1} - L_E$
       $\vee \|\mathsf{w}\|_\infty > \lfloor q/2 \rfloor - L_E \vee \|\mathsf{z}\|_\infty > B - U$
10.   then $j \leftarrow j + 1$ and go to Step 1
11. return $(\mathsf{z}, c)$
```

Fig. 1. Specification of the signature scheme TESLA = (KeyGen, Sign, Verify); for details of the functions checkE and checkS see the explanation of the public parameters and definition of functions.

Public Parameters and Definition of Functions. TESLA is parameterized by the integers n, n', m, α, κ, and the security parameter λ with $m > n > \kappa \geq \lambda$; by the matrix $\mathsf{A} \leftarrow_\$ \mathbb{Z}_q^{m \times n}$; by the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$, by the encoding function $F : \{0,1\}^\kappa \to \mathbb{H}$ (see [31] for more information), by the pseudo-random function $PRF_1 : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^\kappa$, and the pseudo-random generator $PRF_2 : \{0,1\}^\kappa \times \mathbb{Z} \to [-B, B]^n$. The remaining values, *i.e.*, σ, h, d, B, U, q, L, and L_S, are derived as shown in Table 1 and described in Sec. 5.1.

Moreover, we define the functions checkE, introduced in [61, Section 3.2], as follows: for a matrix E, define $\mathsf{E_i}$ to be the i-th row of E. The function $\max_k(\cdot)$ returns the k-th largest entry of a vector. The matrix E is rejected if for any row of E it holds that $\sum_{k=1}^{h} \max_k(\mathsf{E_i})$ is greater than some bound L. We apply a similar check checkS to S: The matrix S is rejected if for any row of S it holds that $\sum_{k=1}^{h} \max_k(\mathsf{S_i})$ is greater than some bound L_S.

Remark 1 (Deterministic signature). Note, that signing is deterministic for each message μ since the randomness is determined by the vector y which is deterministically computed by the secret key and the message to-be-signed. In the original scheme by Bai and Galbraith [8] the vector y was sampled uniformly random in $[-B, B]^n$. As long as we assume that PRF_1 and PRF_2 are pseudo-random functions, the prf-advantage of $PRF_2 \circ PRF_1$ is negligible in the security parameter. Hence, the reduction given in Theorem 2 is tight with our choice of parameters. The idea to use a pseudo-random function to generate signatures deterministically was deployed several times before [11, 13, 34, 48, 59].

Parallel Matrix-Vector Multiplication. The most costly operation during signing is the computation of Ay, which requires nm 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 $nm/8$ cycles per matrix multiplication. This simple lower bound would translate to 254058 cycles for $\mathsf{TESLA\text{-}0}$; however, this is ignoring the fact that each partial product has up to 53 bits (the size of the double-precision mantissa) and requires reduction before accumulation. Reductions are done through a multiplication by an approximate inverse of q, rounding, multiplication by q and subtraction.

Another cause for not even getting close to the lower bound derived from multiply-accumulate throughput is that each coefficient from A needs to be loaded from (at best) L2 cache, because the whole matrix A does not fit into the 32KB of level-1 cache. This was already pointed out as an additional bottleneck by Dagdelen et al. [61]. However, on average, signing computes multiple of those matrix-vector multiplications, all with the constant matrix A. Our software always samples $k = 3$ vectors y, then performs 3 matrix-vector multiplications and then proceeds to investigate whether one of the 3 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 k 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. Computing 3 matrix-vector multiplications of the form Ay with this parallel approach takes 11875355 cycles an average for $\mathsf{TESLA\text{-}0}$.

Signature verification cannot use the k-times-parallel matrix-vector multiplication for the computation of Az. For this task we use an approach similar to [61].

7 Results and Comparison

Table 3 gives benchmarking results for TESLA-0 and TESLA-1, and compares those benchmarks to state-of-the-art results from the literature. Due to the large values q and B of the parameter set TESLA-2, certain elements do no fit into the 53-bit mantissa of a double-precision floating point variable. Hence, we do not compare the performance of TESLA-2 in Table 3.

We obtain our benchmarks on an Intel Core-i7 4770K (Haswell) processor while 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. For all software results we report the sizes of keys and signatures actually produced by the software, not the theoretically smallest possible sizes with full compression.[8]

As can be seen in Table 3, TESLA is several magnitudes faster and sizes are smaller than the only other lattice-based signature scheme that is also proven tightly secure in the quantum random oracle model for the same (classical) security of 96 bits. However, the signature generation and verification algorithms of TESLA-0 are much slower than the implementation of [61] for the same level of security. This is due to the large difference of the parameters chosen, *e.g.*, the matrix dimension m in TESLA-0 is 3156, while $m = 840$ in the parameter set DEG^+ proposed by Dagdelen *et al.* [61]. Note that the parameter set TESLA-0 is chosen according to our security reduction, while the set DEG^+ is not chosen according to the (non-tight) security reduction given in [8].

In the (as of yet quite small) realm of signatures that offer 128 bits of post-quantum security, TESLA-2 offers an alternative to SPHINCS. Public and secret keys of TESLA-2 are much larger than SPHINCS keys, but signatures are several magnitudes smaller. The post-quantum multivariate-based signature scheme Rainbow5640 [22,25] performs best among all listed schemes but unfortunately, comes with no security reduction to its underlying problem.

8 The Gaussian Heuristic

Informally speaking, the *Gaussian heuristic* is a formula asserting that the number of points the intersection $L \cap S$ of a lattice $L \subset \mathbb{R}^m$ and a measurable set $S \subset \mathbb{R}^m$ is approximated by the ratio $\mathrm{vol}(S)/\det(L)$. Use of the Gaussian heuristic has appeared in prior works relating to TESLA [2,4,8]. It seems, however, that a formal statement and proof of the Gaussian heuristic as used in these works has not appeared in any literature. In this section we address this oversight by presenting a formal statement and proof of the Gaussian heuristic for such applications. We emphasize that the present manuscript avoids all use of

[8] We make an exception for BLISS. The authors of the software obviously did not spend any effort on reducing the size of signatures and keys; we report sizes with "trivial" compression through choosing native data types of appropriate sizes.

the Gaussian heuristic, so the results in this section are of independent interest to the rest of this document.

8.1 History

The study of counting lattice points within a bounded region was studied by Gauss in the two-dimensional case, and has been explored extensively in very generalized cases in literature on geometry [24,30,39]. The expectation intuitively follows from Minkowski's theorem, but can deviate arbitrarily for many types of lattices. Due to this fact, the results on general lattices in the literature assert error bounds that are not useful in a cryptographic setting. (For example, the deviation from $\mathrm{vol}(S)/\det(L)$ may be as large as $O(r^{n-1})$ for a set S containing a sphere of radius r).

A more useful bound is presented in [44,45], in terms of the covering radius of a lattice.[9] However, this bound is not defined for sets S whose volume $\mathrm{vol}(S)$ is smaller than the determinant $\det(L)$ of the lattice L. Unfortunately, prior works relating to TESLA seek to apply the Gaussian heuristic to precisely this setting.

Previous applications of the Gaussian heuristic for cryptographic purposes have largely focused on so-called "q-ary lattices", which are lattices L with the property that $q\mathbb{Z}^m \subseteq L \subseteq \mathbb{Z}^m$. In these applications, the set S is a finite subset of $\mathbb{Z}_q^m$ and the lattices L have the form $\Lambda_q(\mathsf{A})$ or $\Lambda_q^{\perp}(\mathsf{A})$ given by

$$\Lambda_q(\mathsf{A}) \overset{\text{def}}{=} \{\mathsf{y} \in \mathbb{Z}^m : \exists \mathsf{x} \in \mathbb{Z}^n \text{ with } \mathsf{y} = \mathsf{A}\mathsf{x}\} \tag{6}$$

$$\Lambda_q^{\perp}(\mathsf{A}) \overset{\text{def}}{=} \left\{\mathsf{y} \in \mathbb{Z}^m : \mathsf{A}^{\mathsf{T}}\mathsf{y} = 0 \pmod q\right\} \tag{7}$$

for arbitrary $\mathsf{A} \in \mathbb{Z}_q^{m\times n}$ with $m \geq n$. We refer to $\Lambda_q^{\perp}(\mathsf{A})$ as the *primal* lattice and $\Lambda_q(\mathsf{A})$ as the *dual* lattice. The following discussion immediately generalizes to *shifted* primal lattices of the form

$$\Lambda_{\mathsf{u},q}^{\perp}(\mathsf{A}) \overset{\text{def}}{=} \left\{\mathsf{y} \in \mathbb{Z}^m : \mathsf{A}^{\mathsf{T}}\mathsf{y} = \mathsf{u} \pmod q\right\} \tag{8}$$

for arbitrary $\mathsf{u} \in \mathbb{Z}_q^n$.

In these applications we have that the volume $\mathrm{vol}(S)$ is simply the number $\#S$ of elements in S. Moreover, if A has full rank ($\mathrm{rank}(A) = n$) then the determinants of the lattices $\Lambda_q(\mathsf{A})$, $\Lambda_{\mathsf{u},q}^{\perp}(\mathsf{A})$ are given by

$$\det(\Lambda_q(\mathsf{A})) = \frac{1}{q^{m-n}} \qquad\qquad \det(\Lambda_{\mathsf{u},q}^{\perp}(\mathsf{A})) = \frac{1}{q^n} \tag{9}$$

The Gaussian heuristic is used in these applications to bound the expectation over a uniformly random A of the size of the intersections $\Lambda_q(\mathsf{A})\cap S$ and $\Lambda_{\mathsf{u},q}^{\perp}(\mathsf{A})\cap S$. Since a random matrix A has full rank except with only negligibly small

[9] Some recent work related to Minkowski's theorem and estimating the covering radius can be found in [55].

probability, the Gaussian heuristic suggests that

$$\operatorname*{Ex}_{\mathsf{A}}\left[\#\left(\varLambda_q(\mathsf{A})\cap S\right)\right]\approx\frac{\#S}{q^{m-n}} \tag{10}$$

$$\operatorname*{Ex}_{\mathsf{A}}\left[\#\left(\varLambda^{\perp}_{\mathsf{u},q}(\mathsf{A})\cap S\right)\right]\approx\frac{\#S}{q^{n}} \tag{11}$$

We provide rigorous statements and proofs of these claims below.

8.2 Proofs of the Gaussian Heuristic for *q*-ary Lattices

Because S is a subset of $\mathbb{Z}_q^m$, the the size of the intersection $\varLambda_q(\mathsf{A})\cap S$ is really a question of the intersection of the image of A with S over the finite field $\mathbb{Z}_q$. Similarly, the intersection $\varLambda^{\perp}_{\mathsf{u},q}(\mathsf{A})\cap S$ is merely the intersection with S of the preimage of u under A over the finite field $\mathbb{Z}_q$.

With this observation in mind, for any vector $\mathsf{u}\in\mathbb{Z}_q^n$ and any matrix $\mathsf{A}\in\mathbb{Z}_q^{m\times n}$ define the following subsets of $\mathbb{Z}_q^m$:

$$\mathrm{Im}(\mathsf{A})\overset{\text{def}}{=}\left\{\mathsf{A}\mathsf{x}:\mathsf{x}\in\mathbb{Z}_q^n,\mathsf{x}\neq 0\right\}, \tag{12}$$

$$\mathrm{PreIm}_{\mathsf{u}}(\mathsf{A})\overset{\text{def}}{=}\left\{\mathsf{y}\in\mathbb{Z}_q^m:\mathsf{A}^{\mathsf{T}}\mathsf{y}=\mathsf{u}\pmod q,\mathsf{y}\neq 0\right\}. \tag{13}$$

Observe that

$$\varLambda_q(\mathsf{A})\cap S=\mathrm{Im}(\mathsf{A})\cap S, \tag{14}$$

$$\varLambda^{\perp}_{\mathsf{u},q}(\mathsf{A})\cap S=\mathrm{PreIm}_{\mathsf{u}}(\mathsf{A})\cap S. \tag{15}$$

We prove the following.

Proposition 3 (Gaussian heuristic, "dual" q-ary lattices). *Let $S\subseteq\mathbb{Z}_q^m$ be any set. It holds that*

$$\operatorname*{Ex}_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\left[\#\left(\mathrm{Im}(\mathsf{A})\cap S\right)\right]\leq\frac{\#S}{q^{m-n}}. \tag{16}$$

Proof. By definition, the expectation in (16) is given by

$$\frac{1}{\#\mathbb{Z}_q^{m\times n}}\sum_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\sum_{\mathsf{v}\in\mathbb{Z}_q^m}\mathrm{bool}\left[\mathsf{v}\in\mathrm{Im}(\mathsf{A})\right]\mathrm{bool}\left[\mathsf{v}\in S\right] \tag{17}$$

$$=\sum_{\mathsf{v}\in\mathbb{Z}_q^m}\mathrm{bool}\left[\mathsf{v}\in S\right]\operatorname*{Pr}_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\left[\mathsf{v}\in\mathrm{Im}(\mathsf{A})\right]. \tag{18}$$

For each v let us bound the probability

$$\operatorname*{Pr}_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\left[\mathsf{v}\in\mathrm{Im}(\mathsf{A})\right]. \tag{19}$$

By definition, the probability (19) is given by

$$\frac{1}{\#\mathbb{Z}_q^{m\times n}}\#\left\{\mathsf{A}\in\mathbb{Z}_q^{m\times n} : \exists \mathsf{x}\in\mathbb{Z}_q^n \text{ with } \mathsf{A}\mathsf{x}=\mathsf{v}, \mathsf{x}\neq 0\right\} \tag{20}$$

$$\leq \frac{1}{\#\mathbb{Z}_q^{m\times n}}\sum_{\substack{\mathsf{x}\in\mathbb{Z}_q^n\\ \mathsf{x}\neq 0}}\#\left\{\mathsf{A} : \mathsf{A}\mathsf{x}=\mathsf{v}\right\}. \tag{21}$$

Observe that

$$\#\left\{\mathsf{A} : \mathsf{A}\mathsf{x}=\mathsf{v}\right\} = q^{mn-m} \tag{22}$$

when $\mathsf{x}\neq 0$ and so the probability (19) is at most

$$\frac{1}{q^{mn}}q^n\cdot q^{mn-m} = \frac{1}{q^{m-n}}. \tag{23}$$

Substituting this bound into (18) we find that the expectation in (16) is at most

$$\sum_{\mathsf{v}\in\mathbb{Z}_q^m}\operatorname{bool}\left[\mathsf{v}\in S\right]\frac{1}{q^{m-n}} = \frac{\#S}{q^{m-n}} \tag{24}$$

as desired.

Proposition 4 (Gaussian heuristic, "primal" q-ary lattices). *Let $S\subseteq\mathbb{Z}_q^m$ be any set and let $\mathsf{u}\in\mathbb{Z}_q^n$ be any vector. It holds that*

$$\operatorname*{Ex}_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\left[\#\left(\operatorname{PreIm}_{\mathsf{u}}(\mathsf{A})\cap S\right)\right] = \frac{\#S}{q^n}. \tag{25}$$

Proof. By definition, the expectation in (25) is given by

$$\frac{1}{\#\mathbb{Z}_q^{m\times n}}\sum_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\sum_{\mathsf{y}\in\mathbb{Z}_q^m}\operatorname{bool}\left[\mathsf{y}\in\operatorname{PreIm}_{\mathsf{u}}(\mathsf{A})\right]\operatorname{bool}\left[\mathsf{y}\in S\right] \tag{26}$$

$$= \sum_{\mathsf{y}\in\mathbb{Z}_q^m}\operatorname{bool}\left[\mathsf{y}\in S\right]\Pr_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\left[\mathsf{y}\in\operatorname{PreIm}_{\mathsf{u}}(\mathsf{A})\right] \tag{27}$$

For each $\mathsf{y}\neq 0$ let us compute probability

$$\Pr_{\mathsf{A}\in\mathbb{Z}_q^{m\times n}}\left[\mathsf{y}\in\operatorname{PreIm}_{\mathsf{u}}(\mathsf{A})\right] = \frac{1}{\#\mathbb{Z}_q^{m\times n}}\#\left\{\mathsf{A}\in\mathbb{Z}_q^{m\times n} : \mathsf{A}^{\mathsf{T}}\mathsf{y}=\mathsf{u} \pmod q\right\}. \tag{28}$$

Observe that

$$\#\left\{\mathsf{A} : \mathsf{A}^{\mathsf{T}}\mathsf{y}=\mathsf{u} \pmod q\right\} = q^{mn-n} \tag{29}$$

since $\mathsf{y}\neq 0$ and so the probability (28) equals

$$\frac{1}{q^{mn}}q^{mn-n} = \frac{1}{q^n}. \tag{30}$$

Substituting this bound into (27) we find that the expectation in (25) equals

$$\sum_{\mathsf{y}\in\mathbb{Z}_q^m}\operatorname{bool}\left[\mathsf{y}\in S\right]\frac{1}{q^n} = \frac{\#S}{q^n} \tag{31}$$

as desired.

Acknowledgments

We are especially grateful to Peter Schwabe for contributions to our software implementation and to the presentation of the paper. We thank Chris Peikert for pointing out a flaw in previous security reductions for TESLA in the random oracle model. We thank Steven Galbraith and anonymous reviewers for valuable feedback on an earlier version of this manuscript.

This work has been supported by 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 2211-C PhD Scholarship, by Ege University under project 2014-FEN-065, and by CryptoWorks21.

References

1. Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-Secure Signatures from Lossy Identification Schemes. In *EUROCRYPT 2012*, LNCS. Springer, 2012.
2. Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An Efficient Lattice-Based Signature Scheme with Provably Secure Instantiation. In *AFRICACRYPT 2016*, volume 9646 of *LNCS*. Springer, 2016.
3. Martin R. Albrecht, Rachel Player, and Sam Scott. On the Concrete Hardness of Learning with Errors. *Journal of Mathematical Cryptology*, 9, 2015.
4. Erdem Alkim, Nina Bindel, Johannes Buchmann, Özgür Dagdelen, and Peter Schwabe. TESLA: Tightly-Secure Efficient Signatures from Standard Lattices. Cryptology ePrint Archive, Report 2015/755, version 20161117:055833, 2015.
5. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-Quantum Key Exchange – a New Hope. In *25th USENIX Security Symposium*. USENIX Association, 2016.
6. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, 2009.
7. László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.
8. Shi Bai and Steven D. Galbraith. An Improved Compression Technique for Signatures Based on Learning with Errors. In *Topics in Cryptology – CT-RSA 2014*, volume 8366 of *LNCS*. Springer, 2014.
9. Rachid El Bansarkhani and Johannes Buchmann. Improvement and Efficient Implementation of a Lattice-Based Signature Scheme. In *Selected Areas in Cryptography*, volume 8282 of *LNCS*. Springer, 2013.
10. Paulo S. L. M. Barreto, Patrick Longa, Michael Naehrig, Jefferson E. Ricardini, and Gustavo Zanon. Sharper Ring-LWE Signatures. Cryptology ePrint Archive, Report 2016/1026, 2016.
11. George Barwood. Digital Signatures Using Elliptic Curves. message `32f519ad.19609226@news.dial.pipex.com` posted to sci.crypt, 1997. `http://groups.google.com/group/sci.crypt/msg/b28aba37180dd6c6`.

12. Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and Weaknesses of Quantum Computing. *SIAM J. Comput.*, 26(5), 1997.
13. Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-Speed High-Security Signatures. In *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *LNCS*. Springer, 2011.
14. 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 *EUROCRYPT 2015*, volume 9056 of *LNCS*. Springer, 2015.
15. Daniel J. Bernstein and Tanja Lange. eBACS: ECRYPT Benchmarking of Cryptographic Systems. `http://bench.cr.yp.to` (accessed 2015-05-19).
16. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random Oracles in a Quantum World. In *ASIACRYPT 2011*, volume 7073 of *LNCS*. Springer, 2011.
17. Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. In *CCS 2016*. ACM, 2016.
18. Xavier Boyen and Qinyi Li. Towards Tightly Secure Lattice Short Signature and Id-Based Encryption. In *ASIACRYPT 2016*, volume 10032 of *LNCS*. Springer, 2016.
19. Michel Boyer, Gilles Brassard, Peter Hoeyer, and Alain Tapp. Tight bounds on quantum searching, 1996.
20. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing (STOC 2013)*, pages 575–584. ACM, 2013.
21. Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Another Look at Tightness II: Practical Issues in Cryptography. Cryptology ePrint Archive, Report 2016/360, 2016.
22. 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 *CHES 2009*, volume 5747 of *LNCS*. Springer, 2009.
23. Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. In *ASIACRYPT 2011*, volume 7073 of *LNCS*. Springer, 2011.
24. Pete L. Clark. Geometry of numbers with applications to number theory, 2012. `http://math.uga.edu/~pete/geometryofnumbers.pdf`.
25. Jintai Ding and Dieter Schmidt. Rainbow, a New Multivariable Polynomial Signature Scheme. In *Applied Cryptography and Network Security*, volume 3531 of *LNCS*. Springer, 2005.
26. Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice Signatures and Bimodal Gaussians. In *CRYPTO 2013*, volume 8042 of *LNCS*. Springer, 2013.
27. Léo Ducas. Accelerating Bliss: the Geometry of Ternary Polynomials. Cryptology ePrint Archive, Report 2014/874, 2014.
28. Edward Eaton and Fang Song. Making Existential-Unforgeable Signatures Strongly Unforgeable in the Quantum Random-Oracle Model. In *10th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2015*, 2015.

29. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for Hard Lattices and New Cryptographic Constructions. In *STOC 2008*. ACM, 2008.
30. P.M. Gruber and J.M. Wills. *Handbook of Convex Geometry*. Number v. 2 in Convex Geometry. North-Holland, 1993.
31. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems. In *CHES 2012*, volume 7428 of *LNCS*. Springer, 2012.
32. Tim Güneysu, Tobias Oder, Thomas Pöppelmann, and Peter Schwabe. Software Speed Records for Lattice-Based Signatures. In *Post-Quantum Cryptography*, volume 7932 of *LNCS*. Springer, 2013.
33. Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, August 1987.
34. Jonathan Katz and Nan Wang. Efficiency Improvements for Signature Schemes with Tight Security Reductions. In *CCS 2003*. ACM, 2003.
35. Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An Introduction to Quantum Computing*. Oxford University Press, Inc., New York, NY, USA, 2007.
36. Neal Koblitz and Alfred Menezes. Another Look at "Provable Security". II. In *INDOCRYPT 2006*, volume 4329 of *LNCS*. Springer, 2006.
37. Neal Koblitz and Alfred Menezes. The Random Oracle Model: a Twenty-Year Retrospective. *Designs, Codes and Cryptography*, 77(2), 2015.
38. Thijs Laarhoven, Michele Mosca, and Joop van de Pol. Finding Shortest Lattice Vectors Faster Using Quantum Search. *Designs, Codes and Cryptography*, 2015.
39. Mordechay B. Levin. On the lower bound in the lattice point remainder problem for a parallelepiped, 2013.
40. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In *Topics in Cryptology – CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, 2011.
41. Mingjie Liu and PhongQ. Nguyen. Solving BDD by enumeration: An update. In *Topics in Cryptology – CT-RSA 2013*, volume 7779 of *LNCS*, pages 293–309. Springer, 2013.
42. Vadim Lyubashevsky. Lattice Signatures without Trapdoors. In *EUROCRYPT 2012*, volume 7237 of *LNCS*. Springer, 2012.
43. Daniele Micciancio. Improving lattice based cryptosystems using the Hermite normal form. In *Cryptography and Lattices*, volume 2146 of *LNCS*, pages 126–145. Springer, 2001.
44. Daniele Micciancio. Almost perfect lattices, the covering radius problem, and applications to Ajtai's connection factor. *SIAM Journal on Computing*, 34(1):118–169, 2004. Preliminary version in STOC 2002.
45. Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, March 2002.
46. Daniele Micciancio and Michael Walter. Fast Lattice Point Enumeration with Minimal Overhead. In *SODA 2015*. SIAM, 2015.
47. Ashley Montanaro. Quantum Walk Speedup of Backtracking Algorithms. arXiv preprint arXiv:1509.02374, 2016.
48. David M'Raïhi, David Naccache, David Pointcheval, and Serge Vaudenay. Computational Alternatives to Random Number Generators. In *Selected Areas in Cryptography*, volume 1556 of *LNCS*. Springer, 1998.
49. Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, New York, 2000.

50. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the forty-first annual ACM symposium on Theory of computing (STOC 2009)*, pages 333–342. ACM, 2009.
51. Chris Peikert. A Decade of Lattice Cryptography. Cryptology ePrint Archive, Report 2015/939, 2015.
52. Chris Peikert and Brent Waters. Lossy Trapdoor Functions and Their Applications. In *STOC 2008*. ACM, 2008.
53. David Pointcheval and Jacques Stern. Security Proofs for Signature Schemes. In *EUROCRYPT 1996*, volume 1070 of *LNCS*. Springer, 1996.
54. Oded Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In *STOC 2005*. ACM, 2005.
55. Oded Regev and Noah Stephens-Davidowitz. A reverse minkowski theorem, 2016.
56. Markus Schmidt. Estimation of the Hardness of the Learning with Errors Problem with a Restricted Number of Samples. GitHub at `https://bitbucket.org/Ma_Schmidt/lwe-estimator`, 2017.
57. Markus Schmidt and Nina Bindel. Estimation of the Hardness of the Learning with Errors Problem with a Restricted Number of Samples. Cryptology ePrint Archive, Report 2017/140, 2017.
58. Dominique Unruh. Quantum Position Verification in the Random Oracle Model. In *CRYPTO 2014*, volume 8617 of *LCNS*. Springer, 2014.
59. John Wigley. Removing need for RNG in Signatures. message `5gov5dpad@wapping.ecs.soton.ac.uk` posted to sci.crypt, 1997. `http://groups.google.com/group/sci.crypt/msg/a6da45bcc8939a89.`
60. Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, 2012.
61. Özgür Dagdelen, Rachid El Bansarkhani, Florian Göpfert, Tim Güneysu, Tobias Oder, Thomas Pöppelmann, Ana Helena Sánchez, and Peter Schwabe. High-Speed Signatures from Standard Lattices. In *LATINCRYPT 2014*, volume 8895 of *LNCS*. Springer, 2015.

Table 3. Overview of state-of-the-art post-quantum signature schemes; signature sizes are given in byte [B], key sizes are given in kilo byte [KB]; 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

Scheme/ Software	Comp. Assum.	ROM? Tight?	QROM? Tight?	Security (PreQ)	 (PostQ)	CPU	Key Size [KB]	Sig. Size [B]	Cycle counts
Selected signature schemes over standard lattices									
GPV [9, 29]	SIS	yes yes	yes yes	96	59	AMD Opteron 8356 (Barcelona)	vk: 27,840 sk: 12,064	30,105	sign: 312,800,000 verify: 50,600,000
DEG$^+$ [8, 61]	SIS, LWE	yes no	–	96	?	Intel Core i7-4770K (Haswell)	vk: 1,581 sk: 891	1,495	sign: 1,203,924 verify: 335,072
TESLA-0 (this paper)	LWE	yes yes	yes yes	96	?	Intel Core-i7-4770K (Haswell)	vk: 4,808 sk: 2,895	1,964	sign: 27,243,747 verify: 5,374,884
TESLA-1 (this paper)	LWE	yes yes	yes yes	128	?	Intel Core-i7-4770K (Haswell)	vk: 11,653 sk: 6,769	2,444	sign: 143,402,231 verify: 19,284,672
Selected signatures schemes over ideal lattices									
GPV-poly [9, 29]	R-SIS	yes yes	yes yes	96	59	AMD Opteron 8356 (Barcelona)	vk: 55 sk: 26	32,972	sign: 80,500,000 verify: 11,500,000
GLP [31, 32, 61][a]	DCK	yes no	–	75–80	?	Intel Core i5-3210M (Ivy Bridge)	vk: 1.5 sk: 0.25	1,186	sign: 452,223 verify: 34,004
Bliss-bI [26, 27][b]	R-SIS, NTRU	yes no	–	128	?	"Intel Core @ 3.4GHz"	vk: 7 sk: 2	1,559	sign: ≈358,400 verify: 102,000
Selected other post-quantum signature schemes									
SPHINCS-256 [14]	Hash collisions 2nd preimage	yes no[c]	yes no[c]	>128	128	Intel Xeon E3-1275 (Haswell)	vk: 1 sk: 1	41,000	sign: 51,636,372 verify: 1,451,004
Rainbow5640 [22, 25][d]	MQ, EIP[e]	–	–	80	?	Intel Xeon E3-1275 (Haswell)	vk: 43 sk: 84	37	sign: 42,700 verify: 36,072

[a] In the benchmarks we include the improvements by Dagdelen et al. presented in [61].

[b] We report sizes of keys and signatures with "trivial" compression as explained in the text.

[c] The security of SPHINCS is reduced tightly from the hardness of finding hash collisions and non-tightly from the hardness of finding 2nd preimages in the standard model. Hence the reduction also holds in the ROM and QROM.

[d] Benchmark on Haswell CPU from [15].

[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.

A Overview of Security Proof

As in Section 4, let $\mathcal{F}$ be a forging algorithm that, on input a TESLA public key (A, T), makes no more than q_h quantum queries to a hash oracle $\mathrm{H}(\cdot)$ and no more than q_s classical queries to a TESLA sign oracle for public key (A, T). Let $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$ denote the probability that $\mathcal{F}$ produces a valid TESLA forgery. We build a LWE-solver $\mathcal{S}$ whose run time is close to that of $\mathcal{F}$ and who solves LWE with success bias close to $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$. It then follows from the presumed hardness of LWE that $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$ must be small.

The LWE-solver $\mathcal{S}$ is described in Algorithm 4. Classical queries made by $\mathcal{F}$ to the sign oracle are simulated by $\mathcal{S}$ as specified in Simulated sign (Algorithm 5). Quantum queries made by $\mathcal{F}$ to the hash oracle are simulated by $\mathcal{S}$ according to the construction of Zhandry based on $2q_h$-wise independent functions [60]. (Alternately, if performance of the simulator is a concern then one could instead use a quantum-resistant pseudorandom function.)

Algorithm 4 LWE-solver $\mathcal{S}$ using a forger $\mathcal{F}$.

```
Input: A LWE instance $(\mathsf{A}, \mathsf{T})$.
Output: “Yes” or “no”.
1: Invoke the forger $\mathcal{F}$ with public key $(\mathsf{A}, \mathsf{T})$.
   Whenever $\mathcal{F}$ makes a hash or sign query, simulate that query as follows:
   Classical sign queries. Execute Simulated sign (Algorithm 5).
   Quantum hash queries. Apply a qauntum circuit that implements a random but fixed $2q_h$-wise independent function, except on inputs that have been re-programmed by Simulated sign.
2: Eventually, $\mathcal{F}$ produces a purported forgery.
   If that forgery is legit then output “yes”, otherwise output “no”.
```

Algorithm 5 Simulated sign

```
Input: Message $\mu$, public key $(\mathsf{A}, \mathsf{T})$.
Output: Signature $(\mathsf{z}, \mathsf{c})$.
1: Choose $\mathsf{z}$ uniformly at random among $(B - U)$-short vectors from $\mathbb{Z}_q^n$.
2: Choose $\mathsf{c} \in \mathbb{H}$ uniformly at random.
3: If $\mathsf{Az} - \mathsf{Tc}$ is not well-rounded then retry at step 1.
4: Re-program the hash oracle $\mathrm{H}(\cdot)$ so that $\mathrm{H}([\mathsf{Az} - \mathsf{Tc}], \mu) = \mathsf{c}$.
5: Return $(\mathsf{z}, \mathsf{c})$.
```

That $\mathcal{S}$ solves LWE with success bias close to $\Pr[\text{forge}(\mathsf{A}, \mathsf{T})]$ is a consequence of the following facts, which are proven in subsequent sections:

Appendix B: For yes-instances of LWE, the probability with which $\mathcal{S}$ outputs "yes" is close to $\Pr[\text{forge}(\mathsf{A},\mathsf{T})]$.

Appendix C: For no-instances of LWE, $\mathcal{F}$ successfully forges (and hence $\mathcal{S}$ outputs "yes") with only negligible probability.

A.1 Notation and Sizes for Various Sets of Vectors

Discussion in appendices B, C, D refers to the following sets of integer vectors:

$\mathbb{Y}$: The set of vectors $\mathsf{y} \in \mathbb{Z}_q^n$ such that y is B-short.
$\Delta\mathbb{Y}$: $\{\mathsf{y} - \mathsf{y}' : \mathsf{y}, \mathsf{y}' \in \mathbb{Y}\}$.
$\mathbb{S}$: The set of vectors $\mathsf{z} \in \mathbb{Z}_q^n$ such that z is $(B-U)$-short.
$\Delta\mathbb{S}$: $\{\mathsf{z} - \mathsf{z}' : \mathsf{z}, \mathsf{z}' \in \mathbb{S}\}$.
$\mathbb{H}$: The set of vectors $c \in \{-1,0,1\}^{n'}$ with exactly h nonzero entries.
$\Delta\mathbb{H}$: $\{\mathsf{c} - \mathsf{c}' : \mathsf{c}, \mathsf{c}' \in \mathbb{H}\}$.
$\mathbb{W}$: The set $\{[\mathsf{w}] : \mathsf{w} \in \mathbb{Z}_q^m\}$ of integer vectors obtained from the high bits of a vector in $\mathbb{Z}_q^m$.
$\Delta\mathbb{L}$: $\left\{\mathsf{x} - \mathsf{x}' : \mathsf{x}, \mathsf{x}' \in \mathbb{Z}_q^m \text{ and } [\mathsf{x}] = [\mathsf{x}']\right\} = [-(2^d-1), 2^d-1]^m$.

The sizes of some of these sets are listed below:

$$\#\mathbb{Y} = (2B-1)^n \qquad \#\Delta\mathbb{Y} = (4B-1)^n \tag{32}$$
$$\#\mathbb{S} = (2(B-U)-1)^n \qquad \#\Delta\mathbb{S} = (4(B-U)-1)^n \tag{33}$$
$$\#\mathbb{H} = \binom{n'}{h} 2^h \tag{34}$$
$$\#\Delta\mathbb{L} = (2^{d+1}-1)^m \tag{35}$$

The size of $\Delta\mathbb{H}$ is computed as follows.

Lemma 1 (Size of $\Delta\mathbb{H}$).

$$\#\Delta\mathbb{H} = \sum_{k=0}^{h} \sum_{i=0}^{h-k} \binom{n'}{2i} 2^{2i} \binom{n'-2i}{k} 2^k \tag{36}$$

Proof. For each $k = 0, \ldots, h$ let $\Delta\mathbb{H}_k \subset \Delta\mathbb{H}$ denote the set of vectors in $\Delta\mathbb{H}$ with exactly k entries in $\{-2, 2\}$ and exactly $n' - k$ entries in $\{-1, 0, 1\}$. Observe that $\#\Delta\mathbb{H} = \sum_{k=0}^{h} \#\Delta\mathbb{H}_k$.

Fix k and for each $i = 0, \ldots, h-k$ let $\Delta\mathbb{H}_{k,i} \subset \Delta\mathbb{H}_k$ denote the set of vectors in $\Delta\mathbb{H}_k$ with exactly $2i$ entries in $\{-1, 1\}$. Observe that $\#\Delta\mathbb{H}_k = \sum_{i=0}^{h-k} \#\Delta\mathbb{H}_{k,i}$.

One can count the number of elements in each $\Delta\mathbb{H}_{k,i}$ as

$$\#\Delta\mathbb{H}_{k,i} = \binom{n'}{2i} 2^{2i} \binom{n'-2i}{k} 2^k, \tag{37}$$

from which the lemma follows.

B Yes-Instances of LWE

In this section we establish a lower bound on the probability

$$\Pr\left[\mathcal{S} \text{ output “yes”} \mid (\mathsf{A},\mathsf{T}) \text{ yes-instance of LWE}\right] \tag{38}$$

in terms of $\Pr\left[\text{forge}(\mathsf{A},\mathsf{T})\right]$. This is accomplished by proving that the simulated oracles are indistinguishable from the real oracles.

To this end we consider an arbitrary distinguisher $\mathcal{D}$ who, like the forger $\mathcal{F}$, makes at most q_h quantum queries to the hash oracle and at most q_s classical queries to the sign oracle. Unlike $\mathcal{F}$, however, $\mathcal{D}$'s goal is merely to distinguish the real oracles from the simulated oracles.

B.1 Adaptively Chosen Queries

An arbitrary distinguisher could adaptively and arbitrarily interleave its hash and sign queries. To facilitate our analysis we wish to model the distinguisher in such a way that sign queries occur at fixed, predictable points throughout the protocol. This goal is accomplished by a continuous accounting method for hash queries that we describe presently.

Standard formalism specifies that a quantum oracle for $\mathrm{H} : X \to Y$ be implemented by a unitary channel $|x\rangle|y\rangle \mapsto |x\rangle|y + \mathrm{H}(x)\rangle$. We modify this formalism so that the unitary channel for $\mathrm{H}(\cdot)$ is a *controlled* unitary channel. Specifically, the channel acts on an additional qubit, the state of which dictates whether the unitary channel is applied:

$$|\mathrm{off}\rangle|x\rangle|y\rangle \mapsto |\mathrm{off}\rangle|x\rangle|y\rangle \tag{39}$$

$$|\mathrm{on}\rangle|x\rangle|y\rangle \mapsto |\mathrm{on}\rangle|x\rangle|y + \mathrm{H}(x)\rangle \tag{40}$$

Consider a new type of distinguisher with the following properties:

1. The distinguisher makes $q_h q_s$ hash queries instead of just q_h hash queries.
2. Exactly one sign query occurs after every q_h hash queries.
3. For each choice of hash oracle $\mathrm{H}(\cdot)$, the distinguisher's total “query magnitude” (in the BBBV sense - see Section B.6) on query states with the control qubit set to $|\mathrm{on}\rangle$ over all $q_h q_s$ hash queries does not exceed q_h.

A distinguisher of this form is called a *live-switch distinguisher*. For later convenience, we refer to the query magnitude on states with the control qubit set to $|\mathrm{on}\rangle$ as the *query magnitude on the live-switch.*

Intuition: a live-switch distinguisher can make “partial” queries to the hash oracle. If it's query state has only α amplitude on the live-switch then the distinguisher is “charged” for only a a $|\alpha|^2$-fraction of a query.

It is clear that any ordinary distinguisher who makes q_h hash queries and q_s sign queries, adaptively chosen and interleaved, could be simulated by a live-switch distinguisher. Thus, live-switch distinguishers are at least as powerful as ordinary distinguishers, and possibly more powerful. We will prove indistinguishability against a live-switch distinguisher, which is more security than is strictly necessary.

The benefit of the live-switch distinguisher is that sign queries occur at fixed points throughout the protocol - one sign query after every q_h hash queries. This property allows us to partition the interaction into q_s *blocks*. Each block consists of q_h quantum queries to the hash oracle, followed by a single classical query to the sign oracle. We prove security of each block and then claim security of the entire interaction inductively.

B.2 The Distinguisher's State, a First Look

To begin, consider the state of $\mathcal{D}$'s system immediately prior to the sign oracle query in the first block. At this point in the interaction the real and simulated oracles are perfectly identical - both respond to the first q_h hash queries in accordance with some fixed choice of hash oracle $\mathrm{H}(\cdot)$. Let ρ_{H} denote the state of $\mathcal{D}$'s system at this point in the interaction, conditioned on $\mathrm{H}(\cdot)$. The sign oracle (both real and simulated) acts as follows on $\mathcal{D}$'s system:

1. Measure the message register, resulting in outcome μ.
2. Select a signature (z, c) for message μ.
3. Prepare an output register in the classical basis state $|\mu\rangle|(\mathsf{z}, \mathsf{c})\rangle$.

These actions can be viewed as a quantum channel. If the sign oracle is a real sign oracle then the signature (z, c) is a function of private randomness and the hash oracle $\mathrm{H}(\cdot)$. In this case, the channel is denoted $\Psi_{\mathrm{real,H}}$. If the sign oracle is a simulated sign oracle then the signature (z, c) is a function only of private randomness. In this case, the channel is denoted Ψ_{sim}.

Thus, the state of $\mathcal{D}$'s system at the end of the first block, conditioned on the choice of $\mathrm{H}(\cdot)$, is either $\Psi_{\mathrm{sim}}(\rho_{\mathrm{H}})$ or $\Psi_{\mathrm{real,H}}(\rho_{\mathrm{H}})$. We will argue that the state $\Psi_{\mathrm{sim}}(\rho_{\mathrm{H}})$ is δ-close to a probabilistic mixture over re-programmed hash oracles $\mathrm{H}'(\cdot)$ of states of the form $\Psi_{\mathrm{real,H'}}(\rho_{\mathrm{H'}})$.

This δ-closeness is preserved by the hash queries in the second block of the interaction, since both the real and simulated hash oracles remain consistent with $\mathrm{H}'(\cdot)$ in this block. Let $\rho_{2,\mathrm{H'}}$ denote the state of $\mathcal{D}$'s system immediately prior to the sign oracle query in the second block. As above, we have that $\Psi_{\mathrm{sim}}(\rho_{2,\mathrm{H'}})$ is δ-close to a mixture of states of the form $\Psi_{\mathrm{real,H''}}(\rho_{2,\mathrm{H''}})$.

Continuing inductively, we see that the state of $\mathcal{D}$'s system at the end of an interaction with simulated oracles is $q_s\delta$-close to a probailistic mixture over hash oracles of states of $\mathcal{D}$'s system at the end of an interaction with real oracles. Averaging over the choice of initial hash oracle $\mathrm{H}(\cdot)$, we then see that the simulated oracles are indistinguishable from the real oracles.

We formalize these arguments in subsequent sections.

B.3 Mid-Sign

Consider the sign oracle Mid-sign of Algorithm 6. Mid-sign should be viewed

Algorithm 6 Mid-sign

Input: Message μ, public key (A, T), secret key (S, E).
Output: Signature (z, c).

```
1: Choose $(\mathsf{y}, \mathsf{c}) \in \mathbb{Y} \times \mathbb{H}$ uniformly at random.
2: $\mathsf{z} \leftarrow \mathsf{y} + \mathsf{Sc}$.
3: If $\mathsf{z} \notin \mathbb{S}$ then retry at step 1.
4: If $\mathsf{Ay} - \mathsf{Ec}$ is not well-rounded then retry at step 1.
5: Re-program the hash oracle $\mathrm{H}(\cdot)$ so that $\mathrm{H}([\mathsf{Ay}], \mu) = \mathsf{c}$.
6: Return $(\mathsf{z}, \mathsf{c})$.
```

as a hybrid of Simulated sign (Algorithm 5) and the real sign oracle Sign (Algorithm 2).

In this section we prove that Mid-sign (Algorithm 6) and Simulated sign (Algorithm 5) are identical. This fact can be stated in terms of quantum channels as follows. Let Ψ_{mid} denote the channel described by Algorithm 6. The claim of this section is that $\Psi_{\mathrm{mid}} = \Psi_{\mathrm{sim}}$.

To this end, define the following sets for each choice of $\mathsf{c} \in \mathbb{H}$:

$$\mathrm{good}_{\mathrm{sim}}(\mathsf{c}) \stackrel{\mathrm{def}}{=} \{\mathsf{z} \in \mathbb{S} : \mathsf{Az} - \mathsf{Tc} \text{ is well-rounded}\} \tag{41}$$
$$\mathrm{good}_{\mathrm{mid}}(\mathsf{c}) \stackrel{\mathrm{def}}{=} \{\mathsf{y} \in \mathbb{Y} : \mathsf{y} + \mathsf{Sc} \in \mathbb{S} \text{ and } \mathsf{Az} - \mathsf{Tc} \text{ is well-rounded}\} \tag{42}$$

We begin with a simple observation on these sets.

Lemma 2. *The mapping* $f : \mathsf{y} \mapsto \mathsf{y} + \mathsf{Sc}$ *is a bijection from* $\mathrm{good}_{\mathrm{mid}}(\mathsf{c})$ *to* $\mathrm{good}_{\mathrm{sim}}(\mathsf{c})$ *with inverse* $f^{-1} : \mathsf{z} \mapsto \mathsf{z} - \mathsf{Sc}$.

Proof. It is clear that $f^{-1}f$ is the identity function on $\mathrm{good}_{\mathrm{mid}}(\mathsf{c})$. It remains to prove the following:

1. For each $\mathsf{y} \in \mathrm{good}_{\mathrm{mid}}(\mathsf{c})$ it holds that $f(\mathsf{y}) \in \mathrm{good}_{\mathrm{sim}}(\mathsf{c})$.
2. For each $\mathsf{z} \in \mathrm{good}_{\mathrm{sim}}(\mathsf{c})$ it holds that $f^{-1}(\mathsf{z}) \in \mathrm{good}_{\mathrm{mid}}(\mathsf{c})$.

To prove item 1 we must show (i) $f(\mathsf{y}) \in \mathbb{S}$, and (ii) $\mathsf{A}f(\mathsf{y}) - \mathsf{Tc}$ is well-rounded. Both items are immediate from the definitions of $\mathrm{good}_{\mathrm{mid}}(\mathsf{c})$ and f.

To prove item 2 we must show (i) $f^{-1}(\mathsf{z}) \in \mathbb{Y}$, and (ii) $\mathsf{A}f^{-1}(\mathsf{z}) - \mathsf{Ec}$ is well-rounded. Item (i) follows from the fact that z is $(B-U)$-short and Sc is U-short. Item (ii) is immediate from the definitions of $\mathrm{good}_{\mathrm{sim}}(\mathsf{c})$ and f^{-1}.

We now prove this section's claim.

Proposition 5 (Equivalence of Mid-sign and Simulated sign). *The observable behaviour of Mid-sign (Algorithm 6) is statistically identical to that of Simulated sign (Algorithm 5). In terms of quantum channels, we have* $\Psi_{\mathrm{mid}} = \Psi_{\mathrm{sim}}$.

Proof. The observable effects of both the Simulated sign and Mid-sign algorithms can be summarized as follows. Given a message μ as input, the algorithm selects (i) a signature $(\mathsf{z}_\mu, \mathsf{c}_\mu)$ as output, and (ii) a vector w_μ inducing a hash input (w_μ, μ) upon which the hash oracle is re-programmed. Thus, to establish statistical equivalence betwen Simulated sign and Mid-sign it suffices to prove that, for each choice of message μ, the joint distribution over $(\mathsf{z}_\mu, \mathsf{c}_\mu, \mathsf{w}_\mu)$ induced by each algorithm is identical.

Fix an arbitrary message μ and let $(Z_{\text{sim}}, C_{\text{sim}}, W_{\text{sim}})$, $(Z_{\text{mid}}, C_{\text{mid}}, W_{\text{mid}})$ denote the joint random variables representing the observable behaviour of Simulated sign and Mid-sign, respectively, on input message μ. We argue that the joint random variables $(Z_{\text{sim}}, C_{\text{sim}})$, $(Z_{\text{mid}}, C_{\text{mid}})$ are identical. The proposition will then follow from the observation that the hash input w to be re-programmed is specified in both algorithms by the same deterministic function of (z, c). Specifically, in Simulated sign we have $\mathsf{w} = [\mathsf{Az} - \mathsf{Tc}]$, whereas in Mid-sign we have $\mathsf{w} = [\mathsf{Ay}]$. Since $\mathsf{Ay} - \mathsf{Ec}$ is well-rounded, we have

$$\mathsf{w} = [\mathsf{Ay}] = [\mathsf{Ay} - \mathsf{Ec}] = [\mathsf{A}(\mathsf{y} + \mathsf{Sc}) - \mathsf{Tc}] = [\mathsf{Az} - \mathsf{Tc}] \tag{43}$$

as desired.

To begin, we argue that

$$\Pr\left[Z_{\text{mid}} = \mathsf{z} \mid C_{\text{mid}} = \mathsf{c}\right] = \Pr\left[Z_{\text{sim}} = \mathsf{z} \mid C_{\text{sim}} = \mathsf{c}\right] \tag{44}$$

for each choice of $\mathsf{c} \in \mathbb{H}$. In Simulated sign, conditioned on a choice of c, the vector z is chosen uniformly among those $\mathsf{z} \in \text{good}_{\text{sim}}(\mathsf{c})$. In Mid-sign, conditioned on a choice of c, the vector y is chosen uniformly among those $\mathsf{y} \in \text{good}_{\text{mid}}(\mathsf{c})$ and the vector z is computed as $\mathsf{z} \leftarrow \mathsf{y} + \mathsf{Sc}$. It follows from Lemma 2 that z is uniform on $\text{good}_{\text{sim}}(\mathsf{c})$, as desired.

Next, we argue that

$$\Pr\left[C_{\text{mid}} = \mathsf{c}\right] = \Pr\left[C_{\text{sim}} = \mathsf{c}\right] \tag{45}$$

for each choice of $\mathsf{c} \in \mathbb{H}$, from which it follows that the joint random variables $(Z_{\text{sim}}, C_{\text{sim}})$, $(Z_{\text{mid}}, C_{\text{mid}})$ are identical. It follows from Lemma 2 that $\#\,\text{good}_{\text{mid}}(\mathsf{c}) = \#\,\text{good}_{\text{sim}}(\mathsf{c})$ for each $c \in \mathbb{H}$. Thus,

$$\Pr\left[C_{\text{mid}} = \mathsf{c}\right] = \frac{\#\,\text{good}_{\text{mid}}(\mathsf{c})}{\sum_{\mathsf{c}'} \#\,\text{good}_{\text{mid}}(\mathsf{c}')} = \frac{\#\,\text{good}_{\text{sim}}(\mathsf{c})}{\sum_{\mathsf{c}'} \#\,\text{good}_{\text{sim}}(\mathsf{c}')} = \Pr\left[C_{\text{sim}} = \mathsf{c}\right] \tag{46}$$

as desired.

B.4 Consistent-Mid-Sign

Broadly speaking, Mid-sign (Algorithm 6) behaves like Sign (Algorithm 2) except that c is selected freshly at random instead of according to some hash oracle $\mathrm{H}(\cdot)$. It is tempting to claim that the only difference between Mid-sign and Sign is that repeated invocations of Sign always use the same hash oracle $\mathrm{H}(\cdot)$, whereas each invocation of Mid-sign switches to another hash oracle $\mathrm{H}'(\cdot)$ that differs from $\mathrm{H}(\cdot)$ on a small number of randomly selected inputs.

However, there is a small probability that the random choices in a given execution of Mid-sign are not consistent with *any* hash oracle. To understand how such an inconsistency can occur, observe that each candidate (y, c) selected by Mid-sign induces an associated claim about the underlying hash oracle: that $\mathrm{H}([\mathsf{Ay}], \mu) = \mathsf{c}$. Suppose Mid-sign rejects one candidate pair (y, c) because $\mathsf{Ay} - \mathsf{Ec}$ is not well-rounded before finally accepting another candidate pair $(\mathsf{y}', \mathsf{c}')$. If $[\mathsf{Ay}] = [\mathsf{Ay}']$ but $\mathsf{c} \neq \mathsf{c}'$ then these two candidates represent conflicting claims about the underlying hash oracle.

To address this problem we present a new sign oracle Consistent-mid-sign in Algorithm 7 and argue that its observable behaviour is negligibly close to that of Mid-sign (Algorithm 6). This fact can be stated in terms of quantum channels as follows. Let $\Psi_{\text{c-mid}}$ denote the channel described by Algorithm 7. The claim of this section is that $\Psi_{\text{c-mid}} \approx \Psi_{\text{mid}}$, meaning that $\Psi_{\text{c-mid}}(\rho) \approx \Psi_{\text{mid}}(\rho)$ for all input states ρ.

Algorithm 7 Consistent-mid-sign

Input: Message μ, public key (A, T), secret key (S, E).
Output: Signature (z, c).

```
1: Initialize the dictionary $\mathcal{A} \subset (\mathbb{W} \mapsto \mathbb{H})$ to the empty dictionary $\mathcal{A} = \emptyset$.
2: Choose $\mathsf{y} \in \mathbb{Y}$ uniformly at random.
3: if $[\mathsf{Ay}] \in \mathcal{A}$ then
4:     $\mathsf{c} \leftarrow \mathcal{A}[[\mathsf{Ay}]]$
5: else
6:     choose $\mathsf{c} \in \mathbb{H}$ uniformly at random
7:     add $\mathcal{A}[[\mathsf{Ay}]] \leftarrow \mathsf{c}$ to the dictionary $\mathcal{A}$.
8: end if
9: $\mathsf{z} \leftarrow \mathsf{y} + \mathsf{Sc}$.
10: If $\mathsf{z} \notin \mathbb{S}$ then retry at step 2.
11: If $\mathsf{Ay} - \mathsf{Ec}$ is not well-rounded then retry at step 2.
12: Re-program the hash oracle $\mathrm{H}(\cdot)$ so that $\mathrm{H}([\mathsf{Ay}], \mu) = \mathsf{c}$.
13: Return $(\mathsf{z}, \mathsf{c})$.
```

The only difference between Consistent-mid-sign and Mid-sign is that each invocation of Consistent-mid-sign remembers the random candidate pairs it selected throughout the invocation and alters them as needed so as to maintain consistency with a hash oracle. Thus, in order to prove $\Psi_{\text{c-mid}} \approx \Psi_{\text{mid}}$ it suffices to prove that only a negligibly small fraction of the random choices made by Mid-sign lead to an inconsistency that is corrected in Consistent-mid-sign.

A sequence $r = \{(\mathsf{y}_i, \mathsf{c}_i)\}_{i=1}^{\infty}$ of random choices made by Mid-sign leads to an inconsistently derived signature only if there exists $k \geq 2$ such that the following conditions hold:

1. $\mathsf{Ay}_1 - \mathsf{Ec}_1, \ldots, \mathsf{Ay}_{k-1} - \mathsf{Ec}_{k-1}$ are not well-rounded.
2. $\mathsf{Ay}_k - \mathsf{Ec}_k$ is well-rounded.

3. $[\mathsf{Ay}_k] \in \{[\mathsf{Ay}_1], \ldots, [\mathsf{Ay}_{k-1}]\}$.

Consider the event that a random sequence r meets conditions 1–3 for some choice of $k \geq 2$, and let inconsistent(r) denote the infinite disjunction of these events over all $k \geq 2$.[10] We seek an upper bound on the probability of event inconsistent(r) over the choice of r. To this end, define the following quantities for each choice of TESLA keys (A, T), (S, E):

nwr(A, E): The probability over $(\mathsf{y}, \mathsf{c}) \in \mathbb{Y} \times \mathbb{H}$ that $\mathsf{Ay} - \mathsf{Ec}$ is not well-rounded.
coll(A, E): The maximum over all $\mathsf{w} \in \mathbb{W}$ of the probability over $(\mathsf{y}, \mathsf{c}) \in \mathbb{Y} \times \mathbb{H}$ that $[\mathsf{Ay} - \mathsf{Ec}] = \mathsf{w}$.

In symbols, these quantities are written

$$\mathrm{nwr}(\mathsf{A}, \mathsf{E}) \stackrel{\mathrm{def}}{=} \Pr_{(\mathsf{y},\mathsf{c}) \in \mathbb{Y} \times \mathbb{H}} [\mathsf{Ay} - \mathsf{Ec} \text{ not well-rounded}] \tag{47}$$

$$\mathrm{coll}(\mathsf{A}, \mathsf{E}) \stackrel{\mathrm{def}}{=} \max_{\mathsf{w} \in \mathbb{W}} \left\{ \Pr_{(\mathsf{y},\mathsf{c}) \in \mathbb{Y} \times \mathbb{H}} [[\mathsf{Ay} - \mathsf{Ec}] = \mathsf{w}] \right\} \tag{48}$$

In Sections B.8 and B.9 we prove bounds on these quantities that hold with high probability over the choice of TESLA keys (A, T), (S, E).

Broadly speaking, nwr(A, E) should be viewed as a constant that's noticeably smaller than 1 - for example, nwr$(\mathsf{A}, \mathsf{E}) = 1/2$. By contrast coll$(\mathsf{A}, \mathsf{E})$ is negligibly small. We prove the following.

Proposition 6 (Probability of inconsistency). *For each choice of TESLA keys* (A, T)*,* (S, E) *it holds that*

$$\Pr_r [\mathrm{inconsistent}(r)] \leq \mathrm{coll}(\mathsf{A}, \mathsf{E}) \frac{\mathrm{nwr}(\mathsf{A}, \mathsf{E})}{(1 - \mathrm{nwr}(\mathsf{A}, \mathsf{E}))^2} \tag{49}$$

Proof. For each $k \geq 2$ the probability with which events 1 and 2 hold is

$$\mathrm{nwr}(\mathsf{A}, \mathsf{E})^{k-1} (1 - \mathrm{nwr}(\mathsf{A}, \mathsf{E})). \tag{50}$$

[10] In item 3 it suffices to look for a collision only between $[\mathsf{Ay}_k]$ and any previous $[\mathsf{Ay}_i]$; we don't need to look for a collision among arbitrary $[\mathsf{Ay}_i] = [\mathsf{Ay}_j]$. This is because such a collision among the bad entries is statistically identical to as if $\mathsf{c}_i = \mathsf{c}_j$. Namely, $[\mathsf{Ay}_i]$ is rejected regardless of whether $\mathsf{c}_i = \mathsf{c}_j$. If however, c_j changes $[\mathsf{Ay}_i]$ from bad to good then that difference will be detected at $k = j$. Thus, there's no need to check for this when $k > j$.

Conditioned on those events, the probability of event 3 is

$$\Pr_{\substack{(\mathsf{y}_1,\mathsf{c}_1),\ldots,(\mathsf{y}_k,\mathsf{c}_k)\notin \mathrm{WR}(\mathsf{A},\mathsf{E})\\(\mathsf{y}_k,\mathsf{c}_k)\in \mathrm{WR}(\mathsf{A},\mathsf{E})}}\left[\bigvee_{i=1}^{k-1}[\mathsf{A}\mathsf{y}_k]=[\mathsf{A}\mathsf{y}_i]\right] \tag{51}$$

$$\leq (k-1)\max_{\mathsf{w}\in\mathbb{W}}\left\{\Pr_{(\mathsf{y},\mathsf{c})\in \mathrm{WR}(\mathsf{A},\mathsf{E})}[[\mathsf{A}\mathsf{y}]=\mathsf{w}]\right\} \tag{52}$$

$$\leq (k-1)\frac{\max_{\mathsf{w}\in\mathbb{W}}\left\{\Pr_{(\mathsf{y},\mathsf{c})\in\mathbb{Y}\times\mathbb{H}}[[\mathsf{A}\mathsf{y}-\mathsf{E}\mathsf{c}]=\mathsf{w}]\right\}}{\Pr_{(\mathsf{y},\mathsf{c})\in\mathbb{Y}\times\mathbb{H}}[\mathsf{A}\mathsf{y}-\mathsf{E}\mathsf{c} \text{ is well-rounded}]} \tag{53}$$

$$=(k-1)\frac{\mathrm{coll}(\mathsf{A},\mathsf{E})}{1-\mathrm{nwr}(\mathsf{A},\mathsf{E})} \tag{54}$$

(Here we have used the notation $(\mathsf{y},\mathsf{c})\in\mathrm{WR}(\mathsf{A},\mathsf{E})$ to mean that $\mathsf{A}\mathsf{y}-\mathsf{E}\mathsf{c}$ is well-rounded.) Thus,

$$\Pr_r[\mathrm{inconsistent}(r)]\leq\sum_{k=2}^{\infty}\mathrm{nwr}(\mathsf{A},\mathsf{E})^{k-1}(1-\mathrm{nwr}(\mathsf{A},\mathsf{E}))(k-1)\frac{\mathrm{coll}(\mathsf{A},\mathsf{E})}{1-\mathrm{nwr}(\mathsf{A},\mathsf{E})} \tag{55}$$

$$=\mathrm{coll}(\mathsf{A},\mathsf{E})\sum_{k=2}^{\infty}(k-1)\,\mathrm{nwr}(\mathsf{A},\mathsf{E})^{k-1}. \tag{56}$$

The proposition then follows from the formula for the derivative of a geometric progression.

An immediate corollary of Proposition 6 is that

$$\|\Psi_{\text{c-mid}}(\rho)-\Psi_{\text{mid}}(\rho)\|_{\mathrm{Tr}}<2\Pr_r[\mathrm{inconsistent}(r)] \tag{57}$$

for all states ρ.

B.5 Consistent-Mid-Sign is a Mixture of Real Sign Oracles

In the previous section we introduced the sign oracle Consistent-mid-sign (Algorithm 7) and claimed that it behaves exactly like Sign (Algorithm 2) with the following exception: repeated invocations of Sign always use the same hash oracle $\mathrm{H}(\cdot)$, whereas each invocation of Consistent-mid-sign switches to another hash oracle $\mathrm{H}'(\cdot)$ that differs from $\mathrm{H}(\cdot)$ on a small fraction of randomly selected inputs.

Let us formalize this claim. Unfortunately, we must introduce some cumbersome notation. For each message μ define the symbols

y_μ: A sequence $\{\mathsf{y}_{\mu,i}\}_{i=1}^{\infty}$ of elements drawn randomly from $\mathbb{Y}$.

$\mathsf{c}_\mu(\mathsf{y}_\mu)$: A sequence $\{\mathsf{c}_{\mu,i}\}_{i=1}^{\infty}$ of elements drawn randomly from $\mathbb{H}$ subject to the constraint that if $[\mathsf{A}\mathsf{y}_{\mu,i}]=[\mathsf{A}\mathsf{y}_{\mu,j}]$ then $\mathsf{c}_{\mu,i}=\mathsf{c}_{\mu,j}$.

The output of Consistent-mid-sign on input μ is a deterministic function of the random data $\mathsf{y}_\mu, \mathsf{c}_\mu(\mathsf{y}_\mu)$. Specifically, let $k(\mu)$ denote the minimum index for which $\mathsf{Ay}_{\mu,k(\mu)} - \mathsf{Ec}_{\mu,k(\mu)}$ is well-rounded. Then Consistent-mid-sign outputs the signature $(\mathsf{y}_{\mu,k(\mu)} + \mathsf{Sc}_{\mu,k(\mu)}, \mathsf{c}_{\mu,k(\mu)})$. For shorthand, write $\tau_\mu = (\mathsf{y}_\mu, \mathsf{c}_\mu(\mathsf{y}_\mu))$.

Let $\mathsf{y} = \{\mathsf{y}_\mu\}_\mu$ and $\mathsf{c}(\mathsf{y}) = \{\mathsf{c}_\mu(\mathsf{y}_\mu)\}_\mu$ denote selections of random data for each possible message μ. For shorthand, write $\tau = (\mathsf{y}, \mathsf{c}(\mathsf{y}))$ so that the behaviour of Consistent-mid-sign on all inputs is completely specified by τ.

For each choice of hash oracle $\mathrm{H}(\cdot)$ and random data τ consider the hash oracle $\mathrm{H}_\tau(\cdot)$ that agrees with $\mathrm{H}(\cdot)$ everywhere except that $\mathrm{H}_\tau([\mathsf{Ay}_{\mu,i}], \mu) = \mathsf{c}_{\mu,i}$ for each message μ and each $i = 1, \ldots, k(\mu)$. In other words,

$$\mathrm{H}_\tau(\mathsf{w}, \mu) = \begin{cases} \mathsf{c}_{\mu,i} & \text{if } \mathsf{w} = [\mathsf{Ay}_{\mu,i}] \text{ for some } i \in \{1, \ldots, k(\mu)\} \\ \mathrm{H}(\mathsf{w}, \mu) & \text{otherwise} \end{cases} . \tag{58}$$

The behaviour of Sign (Algorithm 2) with hash oracle $\mathrm{H}_\tau(\cdot)$ on all inputs is completely specified by y and $\mathrm{H}_\tau(\cdot)$. Moreover, the behaviour of Sign with hash oracle $\mathrm{H}_\tau(\cdot)$ and random data y is *identical* to the behaviour of Consistent-mid-sign with random data τ.

For each choice of random data τ define the following quantum channels:

$\Psi_{\text{c-mid},\tau}$: The quantum channel representing the actions of Consistent-mid-sign (Algorithm 7) with randomness τ.

$\Psi_{\text{real},\mathrm{H}_\tau,\mathsf{y}}$: The quantum channel representing the actions of Sign (Algorithm 2) with hash oracle $\mathrm{H}_\tau(\cdot)$ and randomness y.

The previous observations establish

$$\Psi_{\text{c-mid},\tau} = \Psi_{\text{real},\mathrm{H}_\tau,\mathsf{y}} \tag{59}$$

for each choice of τ. Because $\Psi_{\text{c-mid}}$ is simply a uniform mixture of channels $\Psi_{\text{c-mid},\tau}$, it follows that[11]

$$\Psi_{\text{c-mid}} = \sum_\tau \Pr[\tau]\, \Psi_{\text{real},\mathrm{H}_\tau,\mathsf{y}}. \tag{60}$$

B.6 Re-Programming of Hash Oracles is Hard to Detect

Thus far we have proved that Consistent-mid-sign behaves like a mixture of real sign oracles *when viewed in isolation.* That is, for all states ρ we have

$$\Psi_{\text{c-mid}}(\rho) = \sum_\tau \Pr[\tau]\, \Psi_{\text{real},\mathrm{H}_\tau,\mathsf{y}}(\rho). \tag{61}$$

[11] Strictly speaking, $\Pr[\tau]$ is zero because it represents the uniform distribution over a countably infinite set. We should switch to a probability measure on τ and use an integral instead of a summation over τ. Future versions of this manuscript will address this detail.

But we must extend this proof so that it holds even in the presence of independent information on the underlying hash oracle. In particular, for any hash oracle $\mathrm{H}(\cdot)$ and any state ρ_{H} prepared using only a tractable number of queries to $\mathrm{H}(\cdot)$ we must show that

$$\Psi_{\text{c-mid}}(\rho_{\mathrm{H}}) \approx \sum_{\tau} \Pr[\tau]\, \Psi_{\mathrm{real},\mathrm{H}_\tau,\mathrm{y}}(\rho_{\mathrm{H}_\tau}). \tag{62}$$

To establish this claim it suffices to show that $\rho_{\mathrm{H}} \approx \rho_{\mathrm{H}_\tau}$ with high probability over the choice of τ.

This claim is proven by an application of the BBBV Theorem [12, Theorem 3.3], so let us introduce the formalism necessary to state this theorem. Suppose ρ_{H} was prepared by some party $\mathcal{R}$ using t queries to some hash oracle $\mathrm{H} : X \to Y$. For each $i = 1, \ldots, t$ let ρ_i denote the state of $\mathcal{R}$'s system immediately prior to the ith query to $\mathrm{H}(\cdot)$. For each hash input $x \in X$ let

$$\mathcal{Q}_{\mathcal{R}(\mathrm{H})}(x) \overset{\mathrm{def}}{=} \sum_{i=1}^{t} \mathrm{Tr}\left(|x\rangle\langle x|\rho_i\right) \tag{63}$$

denote the *query magnitude* on input x for $\mathcal{R}$'s interaction with hash oracle $\mathrm{H}(\cdot)$. The BBBV Theorem (or rather, a consequence of it) is as follows.

Theorem 3 ([12, Theorem 3.3]). *The following holds for each $\varepsilon > 0$. Suppose ρ_{H} was prepared by some party $\mathcal{R}$ using t queries to some hash oracle $\mathrm{H} : X \to Y$. Let $\mathrm{H}'(\cdot)$ be a hash oracle that agrees with $\mathrm{H}(\cdot)$ except on a subset $X' \subset X$ of inputs with the property that*

$$\sum_{x \in X'} \mathcal{Q}_{\mathcal{R}(\mathrm{H})}(x) \le \frac{\varepsilon^2}{t}. \tag{64}$$

Let $\rho_{\mathrm{H}'}$ be the state prepared when $\mathcal{R}$ uses hash oracle $\mathrm{H}'(\cdot)$ instead of $\mathrm{H}(\cdot)$. It holds that $\|\rho_{\mathrm{H}'} - \rho_{\mathrm{H}}\|_{\mathrm{Tr}} \le \varepsilon$.

We are now ready to prove the claim of this section.

Proposition 7 (Re-Programming in TESLA). *The following holds for each choice of TESLA keys (A, T), (S, E) and each $\delta > 0$.*

Suppose ρ_{H} was prepared by some party $\mathcal{D}$ using t queries to hash oracle $\mathrm{H}(\cdot)$. Let τ be random data and let $\mathrm{H}_\tau(\cdot)$ be a hash oracle derived from $\mathrm{H}(\cdot)$ and τ as described in Section B.5. Let $\rho_{\mathrm{H}'}$ be the state prepared when $\mathcal{D}$ uses hash oracle $\mathrm{H}'(\cdot)$ instead of $\mathrm{H}(\cdot)$.

Then $\|\rho_{\mathrm{H}_\tau} - \rho_{\mathrm{H}}\|_{\mathrm{Tr}} < \delta$ except with probability at most

$$\frac{t^2}{\delta^2} \frac{\mathrm{coll}(\mathsf{A}, \mathsf{E})}{1 - \mathrm{nwr}(\mathsf{A}, \mathsf{E})} \tag{65}$$

over the choice of τ.

Proof. By Theorem 3 it suffices to prove that the quantity

$$\sum_{\mu} \sum_{i=1}^{k(\mu)} \mathcal{Q}_{\mathcal{D}(\mathrm{H})} \left(\left[\mathsf{Ay}_{\mu,i} \right], \mu \right) \tag{66}$$

is at most δ^2/t with high probability over the choice of τ. To this end, for each message μ let

$$X_\mu = \{ ([\mathsf{Ay}], \mu) : \mathsf{y} \in \mathbb{Y} \} \tag{67}$$

denote the set of hash inputs for message μ that are candidates for re-programming, and let

$$t_\mu = \sum_{x \in X_\mu} \mathcal{Q}_{\mathcal{D}(\mathrm{H})}(x) \tag{68}$$

denote the total query magnitude for message μ. Observe that $t = \sum_\mu t_\mu$.

The quantity (66) is maximized if for each message μ all the query magnitude t_μ alloted to message μ is placed on the element $\mathsf{w} \in \mathbb{W}$ most likely to collide with $[\mathsf{Ay}]$ when $\mathsf{y} \in \mathbb{Y}$ is chosen uniformly at random. In this case, the quantity (66) is at most

$$\sum_\mu k(\mu) t_\mu \operatorname{coll}(\mathsf{A}, \mathsf{E}). \tag{69}$$

By Markov's inequality we have

$$\Pr_{\tau} \left[\sum_\mu k(\mu) t_\mu \operatorname{coll}(\mathsf{A}, \mathsf{E}) \geq \frac{\delta^2}{t} \right] \leq \frac{t}{\delta^2} \operatorname*{Ex}_{\tau} \left[\sum_\mu k(\mu) t_\mu \operatorname{coll}(\mathsf{A}, \mathsf{E}) \right] \tag{70}$$

$$= \frac{t}{\delta^2} \operatorname{coll}(\mathsf{A}, \mathsf{E}) \sum_\mu t_\mu \operatorname*{Ex}_{\tau_\mu} [k(\mu)] \tag{71}$$

Thus, it suffices to bound the expected number $k(\mu)$ of entries one must view from a given list τ_μ before encountering an entry (y, c) for which $\mathsf{Ay} - \mathsf{Ec}$ is well-rounded. We have

$$\operatorname*{Ex}_{\tau_\mu} [k(\mu)] = \sum_{k=1}^{\infty} k \operatorname{nwr}(\mathsf{A}, \mathsf{E})^{k-1} (1 - \operatorname{nwr}(\mathsf{A}, \mathsf{E})) = \frac{1}{1 - \operatorname{nwr}(\mathsf{A}, \mathsf{E})} \tag{72}$$

where the final equality follows from the formula for the derivative of a geometric progression. The proposition follows from $\sum_\mu t_\mu = t$.

B.7 The Distinguisher's State, Revisited

Recall from Section B.2 the state ρ_{H}, which is the state of $\mathcal{D}$'s system immediately prior to the sign query in the first block. Let $\kappa_1 \leq q_h$ denote the query magnitude on the live-switch for the hash oracle in the first block. We proved the following in previous sections:

$$\varPsi_{\text{sim}}(\rho_{\text{H}}) = \varPsi_{\text{mid}}(\rho_{\text{H}}) \tag{73}$$

$$\|\varPsi_{\text{mid}}(\rho_{\text{H}}) - \varPsi_{\text{c-mid}}(\rho_{\text{H}})\|_{\text{Tr}} < 2\,\text{coll}(\mathsf{A},\mathsf{E})\frac{\text{nwr}(\mathsf{A},\mathsf{E})}{(1-\text{nwr}(\mathsf{A},\mathsf{E}))^2} \tag{74}$$

$$\varPsi_{\text{c-mid}}(\rho_{\text{H}}) = \sum_{\tau} \Pr[\tau]\,\varPsi_{\text{real},\text{H}_\tau,\text{y}}(\rho_{\text{H}}) \tag{75}$$

$$\Pr_{\tau}[\|\rho_{\text{H}_\tau} - \rho_{\text{H}}\|_{\text{Tr}} > \varepsilon] < \frac{\kappa_1^2}{\varepsilon^2}\frac{\text{coll}(\mathsf{A},\mathsf{E})}{1-\text{nwr}(\mathsf{A},\mathsf{E})} \tag{76}$$

We conclude that

$$\left\|\varPsi_{\text{sim}}(\rho_{\text{H}}) - \sum_{\tau} \Pr[\tau]\,\varPsi_{\text{real},\text{H}_\tau,\text{y}}(\rho_{\text{H}_\tau})\right\|_{\text{Tr}} < \delta(\kappa_1) \tag{77}$$

where[12]

$$\delta(\kappa) \overset{\text{def}}{=} 2\,\text{coll}(\mathsf{A},\mathsf{E})\frac{\text{nwr}(\mathsf{A},\mathsf{E})}{(1-\text{nwr}(\mathsf{A},\mathsf{E}))^2} + \varepsilon + \frac{\kappa^2}{\varepsilon^2}\frac{\text{coll}(\mathsf{A},\mathsf{E})}{1-\text{nwr}(\mathsf{A},\mathsf{E})}. \tag{78}$$

That is, the state of $\mathcal{D}$'s system at the end of the first block of an interaction with simulated oracles is $\delta(\kappa_1)$-close to a probabilistic mixture over states of $\mathcal{D}$'s system, each of which could have been obtained from an interaction with real hash oracles.

As suggested in Section B.2, we continue inductively throughout the q_s blocks. As with κ_1, let $\kappa_2, \ldots, \kappa_{q_s}$ denote the query magnitude on the live-switch for blocks two through q_s. Define

$$\delta_{\text{yes}} = \sum_{i=1}^{q_s} \delta(\kappa_i) \tag{79}$$

and observe that the state of $\mathcal{D}$'s system at the end of an interaction with simulated oracles is δ_{yes}-close to a probabilistic mixture over states obtained from an interaction with real hash oracles.

Let us compute an upper bound on δ_{yes}. Because each block includes information from hash queries in previous blocks plus one additional hash query learned from the sign oracle, we have

$$\kappa_{i+1} \geq \kappa_i + 1. \tag{80}$$

Because $\mathcal{D}$ is permitted at most q_h total query magnitude on the live-switch for its hash queries, we have

$$\kappa_i \leq q_h + i - 1. \tag{81}$$

[12] The final two terms of (78) are due to the fact that a $\frac{\kappa^2}{\varepsilon^2}\frac{\text{nwr}(\mathsf{A},\mathsf{E})}{(1-\text{nwr}(\mathsf{A},\mathsf{E}))^2}$-fraction of τ lead to a hash oracle $\text{H}_\tau(\cdot)$ for which $\|\rho_{\text{H}} - \rho_{\text{H}_\tau}\|_{\text{Tr}} > \varepsilon$, in which case we assume that $\rho_{\text{H}}, \rho_{\text{H}_\tau}$ are perfectly distinguishable. For all other τ it holds that $\rho_{\text{H}}, \rho_{\text{H}_\tau}$ are ε-close.

It is clear that δ_{yes} is maximized when $\kappa_1 = q_h$, which corresponds to a distinguisher who makes all q_h hash queries before making any of the q_s sign queries. Taking a loose bound $q_h + q_s$ for each κ_i, we obtain

$$\delta_{\text{yes}} < q_s \left(2\,\mathrm{coll}(\mathsf{A},\mathsf{E}) \frac{\mathrm{nwr}(\mathsf{A},\mathsf{E})}{(1-\mathrm{nwr}(\mathsf{A},\mathsf{E}))^2} + \varepsilon + \frac{(q_h+q_s)^2}{\varepsilon^2} \frac{\mathrm{coll}(\mathsf{A},\mathsf{E})}{1-\mathrm{nwr}(\mathsf{A},\mathsf{E})} \right) \quad (82)$$

Finally, because the real and simulated oracles are δ_{yes}-close, it follows that

$$\Pr\left[\mathcal{S} \text{ output ``yes''} \mid (\mathsf{A},\mathsf{T}) \text{ yes-instance of LWE}\right] > \Pr\left[\text{forge}(\mathsf{A},\mathsf{T})\right] - \delta_{\text{yes}} \quad (83)$$

as desired.

B.8 Appendix: Probability of Well-Roundedness

Let ϕ denote the probability that a random vector in $\mathbb{Z}_q^m$ is not well-rounded:

$$\phi \overset{\text{def}}{=} \Pr_{x \in \mathbb{Z}_q^m}\left[x \text{ not well-rounded}\right] \leq m\left(\frac{2L}{2^d} + \frac{2L}{q}\right). \quad (84)$$

The quantity ϕ is a function of the TESLA parameters q, m, d, L. It is a constant that's noticeably smaller than 1.

Recall the definition of $\mathrm{nwr}(\mathsf{A},\mathsf{E})$: for TESLA keys (A,T), (S,E) define $\mathrm{nwr}(\mathsf{A},\mathsf{E})$ as the probability over $(\mathsf{y},\mathsf{c}) \in \mathbb{Y} \times \mathbb{H}$ that $\mathsf{Ay} - \mathsf{Ec}$ is not well-rounded:

$$\mathrm{nwr}(\mathsf{A},\mathsf{E}) \overset{\text{def}}{=} \Pr_{(\mathsf{y},\mathsf{c}) \in \mathbb{Y}\times\mathbb{H}}\left[\mathsf{Ay} - \mathsf{Ec} \text{ not well-rounded}\right]. \quad (85)$$

We prove the following.

Lemma 3 (Probability of well-roundedness). *The following holds for all* $K > 0$. *With probability* $1 - 1/K$ *over the choice of TESLA keys* (A,T), (S,E) *it holds that*

$$\mathrm{nwr}(\mathsf{A},\mathsf{E}) \leq \phi + \sqrt{\frac{K(q+1)}{\#\mathbb{Y}}}. \quad (86)$$

Proof. Our strategy is to bound the variance of $\mathrm{nwr}(\mathsf{A},\mathsf{E})$ over the choice of TESLA keys (A,T), (S,E) and use Chebyshev's inequality. By definition,

$$\mathop{\mathrm{Var}}_{(\mathsf{A},\mathsf{E})}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})\right] = \mathop{\mathrm{Ex}}_{(\mathsf{A},\mathsf{E})}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})^2\right] - \mathop{\mathrm{Ex}}_{(\mathsf{A},\mathsf{E})}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})\right]^2 \quad (87)$$

so it suffices to compute the expectation of $\mathrm{nwr}(\mathsf{A},\mathsf{E})$ and an upper bound on the expectation of $\mathrm{nwr}(\mathsf{A},\mathsf{E})^2$.

We begin by computing the expectation of $\mathrm{nwr}(\mathsf{A},\mathsf{E})$. We have

$$\begin{aligned}
&\operatorname*{Ex}_{(\mathsf{A},\mathsf{E})}[\mathrm{nwr}(\mathsf{A},\mathsf{E})] && (88)\\
&= \sum_{(\mathsf{A},\mathsf{E})} \Pr[(\mathsf{A},\mathsf{E})] \frac{1}{\#\mathbb{Y}\#\mathbb{H}} \sum_{(\mathsf{y},\mathsf{c})} \mathrm{bool}\left[\mathsf{Ay}-\mathsf{Ec} \text{ not well-rounded}\right] && (89)\\
&= \frac{1}{\#\mathbb{Y}\#\mathbb{H}} \sum_{(\mathsf{y},\mathsf{c})} \sum_{(\mathsf{A},\mathsf{E})} \Pr[(\mathsf{A},\mathsf{E})]\, \mathrm{bool}\left[\mathsf{Ay}-\mathsf{Ec} \text{ not well-rounded}\right] && (90)\\
&= \frac{1}{\#\mathbb{Y}\#\mathbb{H}} \sum_{(\mathsf{y},\mathsf{c})} \Pr_{(\mathsf{A},\mathsf{E})}\left[\mathsf{Ay}-\mathsf{Ec} \text{ not well-rounded}\right]. && (91)
\end{aligned}$$

(Here we have used the notation $\mathrm{bool}[s]$ for any statement s that can be either true or false to mean that $\mathrm{bool}[s] = 1$ if the statement is true and $\mathrm{bool}[s] = 0$ otherwise.) So we need to bound the probability

$$\Pr_{(\mathsf{A},\mathsf{E})}\left[\mathsf{Ay}-\mathsf{Ec} \text{ not well-rounded}\right] \tag{92}$$

for each fixed choice of $(\mathsf{y},\mathsf{c}) \in \mathbb{Y}\times\mathbb{H}$. There are two cases:

1. If $\mathsf{y} \neq 0$ then Ay is a uniformly random vector in $\mathbb{Z}_q^m$. So too is $\mathsf{Ay}-\mathsf{Ec}$, since c is fixed and E is independent of A. In this case, the probability (92) equals ϕ.
2. If $\mathsf{y} = 0$ then the probability (92) equals 0, since $-\mathsf{Ec}$ is well-rounded for all E,c.

Case 2 occurs with probability $1/\#\mathbb{Y}$, from which it follows that

$$\operatorname*{Ex}_{(\mathsf{A},\mathsf{E})}[\mathrm{nwr}(\mathsf{A},\mathsf{E})] = \left(1-\frac{1}{\#\mathbb{Y}}\right)\phi. \tag{93}$$

Next, we compute an upper bound on the expectation of $\mathrm{nwr}(\mathsf{A},\mathsf{E})^2$. Similar to the above, we have

$$\operatorname*{Ex}_{(\mathsf{A},\mathsf{E})}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})^2\right] = \frac{1}{(\#\mathbb{Y}\#\mathbb{H})^2} \sum_{(\mathsf{y},\mathsf{c}),(\mathsf{y}',\mathsf{c}')} \Pr_{(\mathsf{A},\mathsf{E})}\left[\mathsf{Ay}-\mathsf{Ec}, \mathsf{Ay}'-\mathsf{Ec}' \text{ not well rounded}\right] \tag{94}$$

and so we need to bound the probability

$$\Pr_{(\mathsf{A},\mathsf{E})}\left[\mathsf{Ay}-\mathsf{Ec}, \mathsf{Ay}'-\mathsf{Ec}' \text{ not well-rounded}\right] \tag{95}$$

for each fixed choice of $(\mathsf{y},\mathsf{c}), (\mathsf{y}',\mathsf{c}') \in \mathbb{Y}\times\mathbb{H}$. There are two cases:

1. If y,y' are nonzero and linearly independent then Ay,Ay' are uniformly random vectors in $\mathbb{Z}_q^m$; so too are $\mathsf{Ay}-\mathsf{Ec}, \mathsf{Ay}'-\mathsf{Ec}'$. In this case, the probability (95) equals ϕ^2.
2. If y,y' are linearly dependent then the probability (95) is at most 1.

Case 2 occurs with probability at most $(q+1)/\#\mathbb{Y}$, from which it follows that

$$\underset{(\mathsf{A},\mathsf{E})}{\mathrm{Ex}}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})^2\right] \leq \left(1 - \frac{q+1}{\#\mathbb{Y}}\right)\phi^2 + \frac{q+1}{\#\mathbb{Y}} \tag{96}$$

Combining these bounds on the expectation of $\mathrm{nwr}(\mathsf{A},\mathsf{E})$ and $\mathrm{nwr}(\mathsf{A},\mathsf{E})^2$ (and employing the inequality $1-(q+1)/\#\mathbb{Y} < (1-1/\#\mathbb{Y})^2$), we obtain the inequality

$$\underset{(\mathsf{A},\mathsf{E})}{\mathrm{Var}}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})\right] \leq \frac{q+1}{\#\mathbb{Y}}. \tag{97}$$

By Chebyshev's inequality it holds that

$$\underset{(\mathsf{A},\mathsf{E})}{\Pr}\left[\left|\mathrm{nwr}(\mathsf{A},\mathsf{E}) - \underset{(\mathsf{A},\mathsf{E})}{\mathrm{Ex}}\left[\mathrm{nwr}(\mathsf{A},\mathsf{E})\right]\right| \geq \sqrt{\frac{K(q+1)}{\#\mathbb{Y}}}\right] \leq \frac{1}{K}. \tag{98}$$

The lemma follows from the expression (93) for the expectation of $\mathrm{nwr}(\mathsf{A},\mathsf{E})$.

B.9 Appendix: Probability of Repetition

Let ψ denote the probability that a random vector $\mathsf{x} \in \mathbb{Z}_q^m$ is in $\Delta\mathbb{L}$:

$$\psi \overset{\mathrm{def}}{=} \underset{\mathsf{x}\in\mathbb{Z}_q^m}{\Pr}\left[\mathsf{x} \in \Delta\mathbb{L}\right] \leq \left(\frac{2^{d+1}}{q}\right)^m. \tag{99}$$

The quantity ψ is a function of the TESLA parameters q, m, d. It is negligibly small.

Recall the definition of $\mathrm{coll}(\mathsf{A},\mathsf{E})$: for TESLA keys (A,T), (S,E) define $\mathrm{coll}(\mathsf{A},\mathsf{E})$ as the maximum over all $\mathsf{w} \in \mathbb{W}$ of the probability over $(\mathsf{y},\mathsf{c}) \in \mathbb{Y}\times\mathbb{H}$ that $[\mathsf{Ay}-\mathsf{Ec}] = \mathsf{w}$:

$$\mathrm{coll}(\mathsf{A},\mathsf{E}) \overset{\mathrm{def}}{=} \max_{\mathsf{w}\in\mathbb{W}}\left\{\underset{(\mathsf{y},\mathsf{c})\in\mathbb{Y}\times\mathbb{H}}{\Pr}\left[[\mathsf{Ay}-\mathsf{Ec}] = \mathsf{w}\right]\right\}. \tag{100}$$

We prove the following.

Lemma 4 (Probability of repetition). *The following holds for all $K > 0$. With probability $1 - 1/K$ over the choice of TESLA keys (A,T), (S,E) it holds that*

$$\mathrm{coll}(\mathsf{A},\mathsf{E}) \leq K\psi. \tag{101}$$

Before proving Lemma 4 let us introduce some notation. Define the set

$$\mathbb{G}(\mathsf{A},\mathsf{E}) \overset{\mathrm{def}}{=} \{(\mathsf{y},\mathsf{c}) \in \Delta\mathbb{Y}\times\Delta\mathbb{H} : \mathsf{Ay}-\mathsf{Ec} \in \Delta\mathbb{L}\}. \tag{102}$$

Some basic facts about the set $\mathbb{G}(\mathsf{A},\mathsf{E})$ are listed below in Lemma 5.

Proof (Proof of Lemma 4). Let $\mathrm{coll}'(\mathsf{A},\mathsf{E})$ denote the probability over $(\mathsf{y},\mathsf{c}) \in \Delta\mathbb{Y} \times \Delta\mathbb{H}$ that $(\mathsf{y},\mathsf{c}) \in \mathbb{G}(\mathsf{A},\mathsf{E})$:

$$\mathrm{coll}'(\mathsf{A},\mathsf{E}) \overset{\mathrm{def}}{=} \Pr_{(\mathsf{y},\mathsf{c})\in\Delta\mathbb{Y}\times\Delta\mathbb{H}} \left[(\mathsf{y},\mathsf{c}) \in \mathbb{G}(\mathsf{A},\mathsf{E})\right]. \tag{103}$$

It follows from Lemma 5 that $\mathrm{coll}'(\mathsf{A},\mathsf{E}) \geq \mathrm{coll}(\mathsf{A},\mathsf{E})$. Thus, it suffices to prove the lemma with $\mathrm{coll}'(\mathsf{A},\mathsf{E})$ in place of $\mathrm{coll}(\mathsf{A},\mathsf{E})$.

Observe that $\mathrm{coll}'(\mathsf{A},\mathsf{E}) \geq 1/\#\Delta\mathbb{Y}$, which follows from the fact that $\#\mathbb{G}(\mathsf{A},\mathsf{E}) \geq \#\Delta\mathbb{H}$ (Lemma 5). Our strategy is to bound the expectation of the positive random variable $\mathrm{coll}'(\mathsf{A},\mathsf{E}) - 1/\#\Delta\mathbb{Y}$ over the choice of TESLA keys (A,T), (S,E) and use Markov's inequality. To this end let us compute the expectation of $\mathrm{coll}'(\mathsf{A},\mathsf{E})$:

$$\begin{aligned}
\mathop{\mathrm{Ex}}_{(\mathsf{A},\mathsf{E})}\left[\mathrm{coll}'(\mathsf{A},\mathsf{E})\right] &= \sum_{(\mathsf{A},\mathsf{E})} \Pr\left[(\mathsf{A},\mathsf{E})\right] \frac{1}{\#\Delta\mathbb{S}\#\Delta\mathbb{H}} \sum_{(\mathsf{y},\mathsf{c})} \mathrm{bool}\left[(\mathsf{y},\mathsf{c}) \in \mathbb{G}(\mathsf{A},\mathsf{E})\right] && (104)\\
&= \frac{1}{\#\Delta\mathbb{S}\#\Delta\mathbb{H}} \sum_{(\mathsf{y},\mathsf{c})} \sum_{(\mathsf{A},\mathsf{E})} \Pr\left[(\mathsf{A},\mathsf{E})\right] \mathrm{bool}\left[(\mathsf{y},\mathsf{c}) \in \mathbb{G}(\mathsf{A},\mathsf{E})\right] && (105)\\
&= \frac{1}{\#\Delta\mathbb{S}\#\Delta\mathbb{H}} \sum_{(\mathsf{y},\mathsf{c})} \Pr_{(\mathsf{A},\mathsf{E})}\left[(\mathsf{y},\mathsf{c}) \in \mathbb{G}(\mathsf{A},\mathsf{E})\right]. && (106)
\end{aligned}$$

So we need to bound the probability

$$\Pr_{(\mathsf{A},\mathsf{E})}\left[(\mathsf{y},\mathsf{c}) \in \mathbb{G}(\mathsf{A},\mathsf{E})\right] \tag{107}$$

for each fixed choice of $(\mathsf{y},\mathsf{c}) \in \Delta\mathbb{Y} \times \Delta\mathbb{H}$. There are two cases:

1. If $\mathsf{y} \neq 0$ then Ay is a uniformly random vector in $\mathbb{Z}_q^m$. So too is $\mathsf{Ay} - \mathsf{Ec}$, since c is fixed and E is independent of A. In this case, the probability (107) is exactly ψ.
2. If $\mathsf{y} = 0$ then the probability (107) is exactly 1.

Case 2 occurs with probability exactly $1/\#\Delta\mathbb{Y}$ over the choice of (y,c). It follows that

$$\mathop{\mathrm{Ex}}_{(\mathsf{A},\mathsf{E})}\left[\mathrm{coll}'(\mathsf{A},\mathsf{E})\right] = \left(1 - \frac{1}{\#\Delta\mathbb{Y}}\right)\psi + \frac{1}{\#\Delta\mathbb{Y}}. \tag{108}$$

Then by Markov's inequality we have

$$\Pr_{(\mathsf{A},\mathsf{E})}\left[\mathrm{coll}'(\mathsf{A},\mathsf{E}) \geq K\left(1 - \frac{1}{\#\Delta\mathbb{Y}}\right)\psi\right] \leq \frac{1}{K}. \tag{109}$$

That is, with probability at least $1 - 1/K$ over the choice of TESLA keys (A,T), (S,E) it holds that

$$\mathrm{coll}'(\mathsf{A},\mathsf{E}) \leq K\left(1 - \frac{1}{\#\Delta\mathbb{Y}}\right)\psi \tag{110}$$

from which the lemma follows.

Lemma 5. *For all TESLA keys* (A,T), (S,E) *and all* $\mathsf{w} \in \mathbb{W}$ *it holds that*

$$\#\mathbb{G}(\mathsf{A},\mathsf{E}) \geq \#\varDelta\mathbb{H} \tag{111}$$

$$\#\mathbb{G}(\mathsf{A},\mathsf{E}) \geq \#\left\{(\mathsf{y},\mathsf{c}) \in \mathbb{Y}\times\mathbb{H} : [\mathsf{Ay}-\mathsf{Ec}] = \mathsf{w}\right\} \tag{112}$$

Proof. The inequality (111) is straightforward: For each $\mathsf{c},\mathsf{c}' \in \mathbb{H}$ we have $[\mathsf{Ec}] = [\mathsf{Ec}'] = [0]$, from which it follows that $(0, \mathsf{c}-\mathsf{c}') \in \mathbb{G}(\mathsf{A},\mathsf{E})$.

It remains to prove the inequality (112). Let $(\mathsf{y},\mathsf{c}),(\mathsf{y}',\mathsf{c}')$ be elements of $\mathbb{Y}\times\mathbb{H}$ with $[\mathsf{Ay}-\mathsf{Ec}] = [\mathsf{Ay}'-\mathsf{Ec}'] = \mathsf{w}$. We claim that $(\mathsf{y}-\mathsf{y}',\mathsf{c}-\mathsf{c}')$ is in $\mathbb{G}(\mathsf{A},\mathsf{E})$. It is clear that $\mathsf{y}-\mathsf{y}' \in \varDelta\mathbb{Y}$ and $\mathsf{c}-\mathsf{c}' \in \varDelta\mathbb{H}$. It remains to verify $\mathsf{A}(\mathsf{y}-\mathsf{y}')-\mathsf{E}(\mathsf{c}-\mathsf{c}') \in \varDelta\mathbb{L}$. We have

$$\mathsf{A}(\mathsf{y}-\mathsf{y}')-\mathsf{E}(\mathsf{c}-\mathsf{c}') = \mathsf{Ay}-\mathsf{Ec}-(\mathsf{Ay}'-\mathsf{Ec}'). \tag{113}$$

Since $\mathsf{Ay}-\mathsf{Ec}$ and $\mathsf{Ay}'-\mathsf{Ec}'$ have the same high bits, it must be that $\mathsf{A}(\mathsf{y}-\mathsf{y}')-\mathsf{E}(\mathsf{c}-\mathsf{c}')$ is the difference of two vectors from $[-(2^{d-1}-1), 2^{d-1}]^m$, from which it follows that $\mathsf{A}(\mathsf{y}-\mathsf{y}')-\mathsf{E}(\mathsf{c}-\mathsf{c}') \in \varDelta\mathbb{L}$.

If $(\mathsf{y}_1,\mathsf{c}_1),\ldots,(\mathsf{y}_k,\mathsf{c}_k)$ are distinct elements of $\mathbb{Y}\times\mathbb{H}$ with $[\mathsf{Ay}_i-\mathsf{Ec}_i] = \mathsf{w}$ for each $i = 1,\ldots,k$ then $(0,0),(\mathsf{y}_1-\mathsf{y}_2,\mathsf{c}_1-\mathsf{c}_2),\ldots,(\mathsf{y}_1-\mathsf{y}_k,\mathsf{c}_1-\mathsf{c}_k)$ must be distinct elements of $\mathbb{G}(\mathsf{A},\mathsf{E})$[13]. We have thus listed k distinct elements of $\mathbb{G}(\mathsf{A},\mathsf{E})$, from which the lemma follows.

C No-Instances of LWE

In this section we prove that the probability

$$\Pr\left[\mathcal{S} \text{ output "yes"} \mid (\mathsf{A},\mathsf{T}) \text{ no-instance of LWE}\right] \tag{114}$$

is small. Our strategy is to identify a correspondence between valid message-signature pairs and "good" inputs to the hash oracle. We then argue that, with high probability over the choice of LWE no-instance (A,T) and hash oracle $\mathrm{H}(\cdot)$, the number of good inputs is a very small fraction of the total number of inputs.

Moreover, whether a given input is good is determined solely by its corresponding output from the hash oracle, implying that the only way to discover good inputs is to perform a search through an unstructured space.

Thus, a computationally bounded forger cannot expect to find good input (and hence a valid forgery), even with quantum access to the random oracle. This argument establishes the claim that the LWE-solver $\mathcal{S}$ outputs "yes" only with small probability, as desired.

C.1 Correspondence Between Valid Signatures and Good Hash Inputs

Let $\mathsf{w} \in \mathbb{W}$, and let μ be an arbitrary message. For any fixed choice of random oracle $\mathrm{H}(\cdot)$ and LWE no-instance (A,T) the hash input (w,μ) is called *good for* $\mathrm{H}(\cdot),\mathsf{A},\mathsf{T}$ if there exists $z \in \mathbb{S}$ with

$$[\mathsf{Az}-\mathsf{T}\,\mathrm{H}(\mathsf{w},\mu)] = \mathsf{w}. \tag{115}$$

[13] By contrast with no-instances, we cannot guarantee that the negations are distinct.

Proposition 8 (Correspondence between valid signatures and good hash inputs). *If* $(\mu, (\mathsf{z}, \mathsf{c}))$ *is a valid message-signature pair for public key* (A, T) *and hash oracle* $\mathrm{H}(\cdot)$ *then* $([\mathsf{Az} - \mathsf{Tc}], \mu)$ *is good for* $\mathrm{H}(\cdot), \mathsf{A}, \mathsf{T}$.

Proof. Write $\mathsf{w} = [\mathsf{Az} - \mathsf{Tc}]$. Because (z, c) is a valid signature for μ we have

$$\mathrm{H}(\mathsf{w}, \mu) = \mathrm{H}([\mathsf{Az} - \mathsf{Tc}], \mu) = \mathsf{c}. \tag{116}$$

Then

$$[\mathsf{Az} - \mathsf{T}\,\mathrm{H}(\mathsf{w}, \mu)] = [\mathsf{Az} - \mathsf{Tc}] = \mathsf{w} \tag{117}$$

as desired.

A corollary of Proposition 8 is that the ability to find a message-signature pair $(\mu, (\mathsf{z}, \mathsf{c}))$ that is valid for public key (A, T) using q_h classical or quantum queries to $\mathrm{H}(\cdot)$ implies the ability to find a hash input (w, μ) that is good for $\mathrm{H}(\cdot), \mathsf{A}, \mathsf{T}$ using the same number of classical or quantum queries to $\mathrm{H}(\cdot)$.

C.2 The fraction of good hash inputs

We wish to bound the probability over hash oracles $\mathrm{H}(\cdot)$ and LWE no-instances (A, T) that a non-negligible fraction of hash inputs (w, μ) are good. To this end, define the sets

$$\mathbb{M} \stackrel{\text{def}}{=} \{(\mathsf{w}, \mu) : \mathsf{w} \in \mathbb{W}, \mu \text{ is a message}\} \tag{118}$$

$$\mathbb{M}(\mathrm{H}, \mathsf{A}, \mathsf{T}) \stackrel{\text{def}}{=} \{(\mathsf{w}, \mu) \in \mathbb{M} : (\mathsf{w}, \mu) \text{ is good for } \mathrm{H}(\cdot), \mathsf{A}, \mathsf{T}\} \tag{119}$$

Discussion is somewhat complicated by the fact that there is an infinite number messages, and hence $\mathbb{M}$ and $\mathbb{M}(\mathrm{H}, \mathsf{A}, \mathsf{T})$ are infinite sets. For ease of exposition we presume a fixed, large upper bound such as 2^{2^λ} on the size of $\mathbb{M}$. After all, no computationally bounded forger could possibly query the hash oracle on inputs whose bit length exceeds 2^λ. Under this presumption, $\mathbb{M}$ is a finite set and so $\#\mathbb{M}$ is a positive integer. The ratio

$$\frac{\#\mathbb{M}(\mathrm{H}, \mathsf{A}, \mathsf{T})}{\#\mathbb{M}} \tag{120}$$

is the fraction of inputs that are good.

Our goal is to show that the ratio (120) is negligibly small with high probability over the choice of $\mathrm{H}(\cdot), \mathsf{A}, \mathsf{T}$. To this end, for each message $(\mathsf{w}, \mu) \in \mathbb{M}$ define the boolean random variable

$$X_{(\mathsf{w},\mu)} = \begin{cases} 1 & \text{if } (\mathsf{w}, \mu) \text{ is good for } \mathrm{H}(\cdot), \mathsf{A}, \mathsf{T} \\ 0 & \text{otherwise} \end{cases} \tag{121}$$

and observe

$$\frac{\#\mathbb{M}(\mathrm{H}, \mathsf{A}, \mathsf{T})}{\#\mathbb{M}} = \frac{1}{\#\mathbb{M}} \sum_{(\mathsf{w},\mu)\in\mathbb{M}} X_{(\mathsf{w},\mu)}, \tag{122}$$

which is an average over boolean random variables. Moreover, the random variables $X_{(\mathsf{w},\mu)}$ are independent and so we may apply Hoeffding bounds to obtain

$$\Pr_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[\frac{\#\mathbb{M}(\mathrm{H},\mathsf{A},\mathsf{T})}{\#\mathbb{M}} - \operatorname*{Ex}_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[\frac{\#\mathbb{M}(\mathrm{H},\mathsf{A},\mathsf{T})}{\#\mathbb{M}}\right] \geq \delta\right] \leq \exp\left(-2\#\mathbb{M}\delta^2\right). \quad (123)$$

Because $\#\mathbb{M}$ is very large relative to other TESLA parameters, we may choose δ so small that it can safely be assumed to equal zero. For example, if $\#\mathbb{M} = 2^{2^\lambda}$ then the probability (123) is negligibly small even when δ is as small as $2^{-2^{\lambda-2}}$. Thus, the ratio (120) is almost certain to be very close to its expecation

$$\operatorname*{Ex}_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[\frac{\#\mathbb{M}(\mathrm{H},\mathsf{A},\mathsf{T})}{\#\mathbb{M}}\right]. \quad (124)$$

This expectation equals

$$\frac{1}{\#\mathbb{M}} \sum_{(\mathsf{w},\mu)\in\mathbb{M}} \operatorname*{Ex}_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[X_{(\mathsf{w},\mu)}\right] \quad (125)$$

and by definition,

$$\operatorname*{Ex}_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[X_{(\mathsf{w},\mu)}\right] = \Pr_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[(\mathsf{w},\mu) \text{ is good for } \mathrm{H}(\cdot),\mathsf{A},\mathsf{T}\right]. \quad (126)$$

It remains to bound this probability for each hash input (w,μ).

C.3 Good Hash Inputs are Rare

For each choice of $\mathsf{w} \in \mathbb{W}$ and LWE no-instance (A,T) we define the set $\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T}) \subset \mathbb{H}$ as

$$\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T}) \overset{\text{def}}{=} \{c \in \mathbb{H} \mid \exists z \in \mathbb{S} : [\mathsf{A}z - \mathsf{T}c] = \mathsf{w}\}. \quad (127)$$

Observe that a hash input (w,μ) is good for $\mathrm{H}(\cdot),\mathsf{A},\mathsf{T}$ if and only if $\mathrm{H}(\mathsf{w},\mu) \in \mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})$. Thus,

$$\Pr_{\mathrm{H},(\mathsf{A},\mathsf{T})}\left[(\mathsf{w},\mu) \text{ is good for } \mathrm{H}(\cdot),\mathsf{A},\mathsf{T}\right] = \operatorname*{Ex}_{(\mathsf{A},\mathsf{T})}\left[\frac{\#\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})}{\#\mathbb{H}}\right] \quad (128)$$

We prove the following.

Proposition 9 (Good Hash Inputs are Rare). *For all* $\mathsf{w} \in \mathbb{W}$ *it holds that*

$$\operatorname*{Ex}_{(\mathsf{A},\mathsf{T})}\left[\max_{\mathsf{w}\in\mathbb{W}}\left\{\frac{\#\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})}{\#\mathbb{H}}\right\}\right] \leq \frac{1}{2\#\mathbb{H}}\left(1 + \frac{\#\Delta\mathbb{H}\#\Delta\mathbb{S}\#\Delta\mathbb{L}}{q^m}\right). \quad (129)$$

Proof. Define the set

$$\mathbb{D}(\mathsf{A},\mathsf{T}) \overset{\text{def}}{=} \{\mathsf{b} \in \Delta\mathbb{H} : \exists \mathsf{y} \in \Delta\mathbb{S} \text{ with } \mathsf{A}\mathsf{y} - \mathsf{T}\mathsf{b} \in \Delta\mathbb{L}\}. \quad (130)$$

In Lemma 6 below we prove

$$\#\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T}) \le \frac{\#\mathbb{D}(\mathsf{A},\mathsf{T})+1}{2} \tag{131}$$

for all $\mathsf{w} \in \mathbb{W}$. Thus, it suffices to bound the expectation

$$\operatorname*{Ex}_{(\mathsf{A},\mathsf{T})}\left[\frac{\#\mathbb{D}(\mathsf{A},\mathsf{T})+1}{2\#\mathbb{H}}\right] = \frac{1}{2\#\mathbb{H}}\left(1+\operatorname*{Ex}_{(\mathsf{A},\mathsf{T})}[\#\mathbb{D}(\mathsf{A},\mathsf{T})]\right) \tag{132}$$

We have

$$\operatorname*{Ex}_{(\mathsf{A},\mathsf{T})}[\#\mathbb{D}(\mathsf{A},\mathsf{T})] = \frac{1}{\#(\mathsf{A},\mathsf{T})}\sum_{(\mathsf{A},\mathsf{T})}\#\left\{\mathsf{b}\in\varDelta\mathbb{H} : \exists\mathsf{y}\in\varDelta\mathbb{S}\text{ with }\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}\right\} \tag{133}$$

$$= \frac{1}{\#(\mathsf{A},\mathsf{T})}\sum_{(\mathsf{A},\mathsf{T})}\sum_{\mathsf{b}\in\varDelta\mathbb{H}}\mathrm{bool}\left[\exists\mathsf{y}\in\varDelta\mathbb{S}\text{ with }\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}\right] \tag{134}$$

$$\le \frac{1}{\#(\mathsf{A},\mathsf{T})}\sum_{(\mathsf{A},\mathsf{T})}\sum_{\mathsf{b}\in\varDelta\mathbb{H}}\sum_{\mathsf{y}\in\varDelta\mathbb{S}}\mathrm{bool}\left[\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}\right] \tag{135}$$

$$= \sum_{\mathsf{b}\in\varDelta\mathbb{H}}\sum_{\mathsf{y}\in\varDelta\mathbb{S}}\frac{1}{\#(\mathsf{A},\mathsf{T})}\sum_{(\mathsf{A},\mathsf{T})}\mathrm{bool}\left[\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}\right] \tag{136}$$

$$= \sum_{\mathsf{b}\in\varDelta\mathbb{H}}\sum_{\mathsf{y}\in\varDelta\mathbb{S}}\operatorname*{Pr}_{(\mathsf{A},\mathsf{T})}\left[\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}\right] \tag{137}$$

For each fixed choice of $\mathsf{y}\in\varDelta\mathbb{S}$, $\mathsf{b}\in\varDelta\mathbb{H}$, if A,T are uniformly random matrices then $\mathsf{Ay}-\mathsf{Tb}$ is a uniformly random vector from $\mathbb{Z}_q^m$. Thus, the probability $\Pr_{(\mathsf{A},\mathsf{T})}[\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}]$ is simply the probability that a random vector lands in $\varDelta\mathbb{L}$. That is,

$$\operatorname*{Pr}_{(\mathsf{A},\mathsf{T})}\left[\mathsf{Ay}-\mathsf{Tb}\in\varDelta\mathbb{L}\right] = \frac{\#\varDelta\mathbb{L}}{q^m}. \tag{138}$$

Thus, the expectation becomes

$$\operatorname*{Ex}_{(\mathsf{A},\mathsf{T})}[\#\mathbb{D}(\mathsf{A},\mathsf{T})] \le \sum_{\mathsf{b}\in\varDelta\mathbb{H}}\sum_{\mathsf{y}\in\varDelta\mathbb{S}}\frac{\#\varDelta\mathbb{L}}{q^m} = \frac{\#\varDelta\mathbb{H}\#\varDelta\mathbb{S}\#\varDelta\mathbb{L}}{q^m} \tag{139}$$

as desired.

Lemma 6. *Let* $\mathbb{D}(\mathsf{A},\mathsf{T})$ *be as defined in* (130). *For all LWE no-instances* (A,T) *and all* $\mathsf{w}\in\mathbb{W}$ *it holds that*

$$\#\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T}) \le \frac{\#\mathbb{D}(\mathsf{A},\mathsf{T})+1}{2}. \tag{140}$$

Proof. Let $\mathsf{c},\mathsf{c}'\in\mathbb{H}(\mathsf{w},\mathsf{A},\mathsf{T})$ as witnessed by $\mathsf{z},\mathsf{z}'\in\mathbb{S}$, respectively. We claim that $\mathsf{c}-\mathsf{c}'$ is in $\mathbb{D}(\mathsf{A},\mathsf{T})$. It is clear that $\mathsf{c}-\mathsf{c}'\in\varDelta\mathbb{H}$ and $\mathsf{z}-\mathsf{z}'\in\varDelta\mathbb{S}$. It remains to verify $\mathsf{A}(\mathsf{z}-\mathsf{z}')-\mathsf{T}(\mathsf{c}-\mathsf{c}')\in\varDelta\mathbb{L}$. We have

$$\mathsf{A}(\mathsf{z}-\mathsf{z}')-\mathsf{T}(\mathsf{c}-\mathsf{c}') = \mathsf{Az}-\mathsf{Tc}-(\mathsf{Az}'-\mathsf{Tc}'). \tag{141}$$

Since $\mathsf{Az} - \mathsf{Tc}$ and $\mathsf{Az}' - \mathsf{Tc}'$ have the same high bits, it must be that $\mathsf{A}(\mathsf{z} - \mathsf{z}') - \mathsf{T}(\mathsf{c} - \mathsf{c}')$ is the difference of two vectors from $[-(2^{d-1} - 1), 2^{d-1}]^m$, from which it follows that $\mathsf{A}(\mathsf{z} - \mathsf{z}') - \mathsf{T}(\mathsf{c} - \mathsf{c}') \in \varDelta\mathbb{L}$. A similar argument proves that the negation $\mathsf{c}' - \mathsf{c} \in \mathbb{D}(\mathsf{A}, \mathsf{T})$.

If $\mathsf{c}_1, \ldots, \mathsf{c}_k$ are distinct elements of $\mathbb{H}(\mathsf{w}, \mathsf{A}, \mathsf{T})$ then $0, \mathsf{c}_1 - \mathsf{c}_2, \ldots, \mathsf{c}_1 - \mathsf{c}_k$ must be distinct elements of $\mathbb{D}(\mathsf{A}, \mathsf{T})$. Similarly, the negations $\mathsf{c}_2 - \mathsf{c}_1, \ldots, \mathsf{c}_k - \mathsf{c}_1$ are also distinct elements of $\mathbb{D}(\mathsf{A}, \mathsf{T})$. To see that $\mathsf{c}_1 - \mathsf{c}_2, \ldots, \mathsf{c}_1 - \mathsf{c}_k$ are all distinct from their negations, observe that

$$\mathsf{c}_1 - \mathsf{c}_i = -(\mathsf{c}_1 - \mathsf{c}_j) \implies 2\mathsf{c}_1 = \mathsf{c}_i + \mathsf{c}_j \implies \mathsf{c}_i = \mathsf{c}_j \tag{142}$$

where the final implication follows from the fact that the entries of $\mathsf{c}_1, \mathsf{c}_i, \mathsf{c}_j$ are all in $\{-1, 0, 1\}$. We have thus listed $2k - 1$ distinct elements of $\mathbb{D}(\mathsf{A}, \mathsf{T})$, from which the lemma follows.

C.4 Forgers Cannot Forge on LWE No-Instances

Proposition 9 provides a bound on the fraction δ_{no} of hash inputs that are good. Moreover, since the goodness of a hash input (w, μ) depends solely on whether $\mathrm{H}(\mathsf{w}, \mu)$ is in $\mathbb{H}(\mathsf{w}, \mathsf{A}, \mathsf{T})$, the set of all good hash inputs is a randomly selected set. Thus, the only way to find a good hash input is via search through an unstructured space.

It then follows from lower bounds for quantum search [19] that any algorithm making no more than q_h quantum queries to $\mathrm{H}(\cdot)$ finds a good hash input—and thus a valid TESLA forgery—with probability no larger than

$$2(q_h + 1)\sqrt{\delta_{\text{no}}}. \tag{143}$$

We therefore obtain

$$\Pr\left[\mathcal{S} \text{ output “yes”} \mid (\mathsf{A}, \mathsf{T}) \text{ no-instance of LWE}\right] \le 2(q_h + 1)\sqrt{\delta_{\text{no}}}. \tag{144}$$

D Security: Putting it all Together

Assuming that no algorithm with run time comparable to that of $\mathcal{S}$ can solve LWE with success bias exceeding ε, we have:

$$\begin{aligned} \varepsilon \ge\ & \Pr\left[\mathcal{S} \text{ output “yes”} \mid (\mathsf{A}, \mathsf{T}) \text{ yes-instance of LWE}\right] & (145)\\ & - \Pr\left[\mathcal{S} \text{ output “yes”} \mid (\mathsf{A}, \mathsf{T}) \text{ no-instance of LWE}\right]. & (146) \end{aligned}$$

We know that

$$\Pr\left[\mathcal{S} \text{ output “yes”} \mid (\mathsf{A}, \mathsf{T}) \text{ yes-instance of LWE}\right] \ge \Pr\left[\text{forge}\,(\mathsf{A}, \mathsf{T})\right] - \delta_{\text{yes}}. \tag{147}$$

Against a quantum forger, we have that

$$\Pr\left[\mathcal{S} \text{ output “yes”} \mid (\mathsf{A}, \mathsf{T}) \text{ no-instance of LWE}\right] \le 2(q_h + 1)\sqrt{\delta_{\text{no}}}, \tag{148}$$

implying that

$$\Pr\left[\text{forge}(\mathsf{A},\mathsf{T})\right] \le \delta_{\text{yes}} + 2(q_h+1)\sqrt{\delta_{\text{no}}} + \varepsilon. \tag{149}$$

Against a *classical* forger, we can remove the quadratic speedup on the lower bound query complexity, and the probability becomes

$$\Pr\left[\text{forge}(\mathsf{A},\mathsf{T})\right] \le \delta_{\text{yes}} + q_h \cdot \delta_{\text{no}} + \epsilon. \tag{150}$$

We make some simplifying assumptions on the choice of TESLA parameters. These assumptions are not necessary in order to derive a negligibly small upper bound on the forger's success probability—they merely facilitate a simplified statement of the upper bound.

Definition 3 (Convenient TESLA Parameters). *TESLA parameters are* convenient *if the following bounds hold:*

$$\phi + \sqrt{\frac{2^{\lambda}(q+1)}{\#\mathbb{Y}}} < 1/2 \tag{151}$$

$$\#\Delta\mathbb{H}\#\Delta\mathbb{Y}\#\Delta\mathbb{L} < q^m. \tag{152}$$

All our proposed parameter sets for TESLA meet this condition.

We now incorporate our bounds on δ_{yes} and δ_{no} in order to derive an explicit upper bound on the forger's success probability.

By applying Lemma 3 with $K = 2^{\lambda}$, as well as equation 151 in definition 3, we can note that with probability $1-2^{-\lambda}$ over the choice of (A,E), $\text{nwr}(\mathsf{A},\mathsf{E}) \le 1/2$. Incorporating this into equation 82, we see that

$$\delta_{\text{yes}} \le q_s\gamma + 4q_s\,\text{coll}(\mathsf{A},\mathsf{E})\left(1 + \frac{(q_h+q_s)^2}{2\gamma^2}\right). \tag{153}$$

From lemma 4, we have a bound on $\text{coll}(\mathsf{A},\mathsf{E})$ that holds with probability $1 - 1/K_{\text{coll}}$.

$$\delta_{\text{yes}} \le q_s\gamma + 4q_s\left(\frac{2^{d+1}}{q}\right)^m K_{\text{coll}}\left(1 + \frac{(q_h+q_s)^2}{2\gamma^2}\right) \tag{154}$$

At this point, we note that our result on this bound holds for whatever γ we may choose. As we want the first term to be exponentially small, we will select $\gamma = \frac{1}{2^{\lambda}q_s}$. Then, using the simplification that $1 + \frac{(q_h+q_s)^2}{2\gamma^2} \approx \frac{(q_h+q_s)^2}{2\gamma^2}$ we get

$$\delta_{\text{yes}} \le \frac{1}{2^{\lambda}} + \frac{2^{m(d+1)+2\lambda+1}}{q^m}(q_h+q_s)^2 q_s^3 K_{\text{coll}}. \tag{155}$$

From proposition 9 we have

$$\delta_{\text{no}} \le \frac{1}{2\#\mathbb{H}}\left(1 + \frac{\#\Delta\mathbb{H}\#\Delta\mathbb{S}\#\Delta\mathbb{L}}{q^m}\right). \tag{156}$$

Using equation 152 of Definition 3 we can simplify this bound on δ_{no} to

$$\delta_{\text{no}} \leq \frac{1}{\#\mathbb{H}}. \tag{157}$$

Finally, we substitute this, and our bound for δ_{yes} into (149). We also note that $\#\mathbb{H} = 2^h \binom{n'}{h}$. We also must consider the probability with which our bounds do not hold. Doing this, we get that $\Pr\left[\text{forge}(\mathsf{A}, \mathsf{T})\right]$ is at most

$$\varepsilon + \frac{1}{2^\lambda} + \frac{2^{m(d+1)+2\lambda+1}}{q^m}(q_h + q_s)^2 q_s^3 K_{\text{coll}} + 2(q_h+1)\sqrt{\frac{1}{2^h\binom{n'}{h}}} + \frac{1}{K_{\text{nwr}}} + \frac{1}{K_{\text{coll}}} \tag{158}$$

Then by choosing each K value to be 2^λ, we get that this is equal to

$$\varepsilon + \frac{3}{2^\lambda} + \frac{2^{m(d+1)+3\lambda+1}}{q^m}(q_h + q_s)^2 q_s^3 + 2(q_h+1)\sqrt{\frac{1}{2^h\binom{n'}{h}}}. \tag{159}$$

Classically, we can similarly derive that the adversary's success is bounded by

$$\varepsilon + \frac{3}{2^\lambda} + \frac{2^{m(d+1)+3\lambda+1}}{q^m}(q_h + q_s)^2 q_s^3 + q_h \frac{1}{2^h\binom{n'}{h}} \tag{160}$$