Asymptotically Efficient Lattice-Based Digital Signatures*

Vadim Lyubashevsky and Daniele Micciancio

University of California, San Diego
La Jolla, CA 92093-0404, USA
{vlyubash,daniele}@cs.ucsd.edu

Abstract. We give a direct construction of digital signatures based on the complexity of approximating the shortest vector in ideal (e.g., cyclic) lattices. The construction is provably secure based on the worst-case hardness of approximating the shortest vector in such lattices within a polynomial factor, and it is also asymptotically efficient: the time complexity of the signing and verification algorithms, as well as key and signature size is almost linear (up to poly-logarithmic factors) in the dimension n of the underlying lattice. Since no sub-exponential (in n) time algorithm is known to solve lattice problems in the worst case, even when restricted to cyclic lattices, our construction gives a digital signature scheme with an essentially optimal performance/security trade-off.

1 Introduction

Digital signature schemes, initially proposed in Diffie and Hellman's seminal paper [9] and later formalized by Goldwasser, Micali and Rivest, [15], are among the most important and widely used cryptographic primitives. Still, our understanding of these intriguing objects is somehow limited.

The definition of digital signatures clearly fits within the public key cryptography framework. However, efficiency considerations aside, the existence of secure digital signatures schemes can be shown to be equivalent to the existence of conventional (symmetric) cryptographic primitives like pseudorandom generators, one-way hash functions, private key encryption, or even just one-way functions [23, 27]. There is a big gap, both theoretical and practical, between the efficiency of known constructions implementing public-key and private-key cryptography. In the symmetric setting, functions are often expected to run in time which is linear or almost linear in the security parameter k. However, essentially all known public key encryption schemes with a supporting proof of security are based on algebraic functions that take at least $\Omega(k^2)$ time to compute, where 2^k is the conjectured hardness of the underlying problem. For example, all factoring based schemes must use keys of size approximately $O(k^3)$ to achieve k bits of

* Research supported in part by NSF grant CCF-0634909. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

security to counter the best known sub-exponential time factoring algorithms, and modular exponentiation raises the time complexity to over $\omega(k^4)$ even when restricted to small k-bit exponents and implemented with an asymptotically fast integer multiplication algorithm.

When efficiency is taken into account, digital signatures seem much closer to public key encryption schemes than to symmetric encryption primitives. Most signature schemes known to date employ the same set of number theoretic techniques commonly used in the construction of public key encryption schemes, and result in similar complexity. Digital signatures based on arbitrary one-way hash functions have also been considered, due to the much higher speed of conjectured one-way functions (e.g., instantiated with common block ciphers as obtained from ad-hoc constructions) compared to the cost of modular squaring or exponentiation operations typical of number theoretic schemes. Still, the performance advantage of one-way function is often lost in the process of transforming them into digital signature schemes: constructions of signature schemes from non-algebraic one-way functions almost invariably rely on Lamport and Diffie's [9] one-time signature scheme (and variants thereof) which requires a number of one-way function applications essentially proportional to the security parameter. So, even if the one-way function can be computed in linear time $O(k)$, the complexity of the resulting signature scheme is again at least quadratic $\Omega(k^2)$.

Therefore, a question of great theoretical and practical interest, is whether digital signature schemes can be realized at essentially the same cost as symmetric key cryptographic primitives. While a generic construction that transforms any one-way function into a signature scheme with similar efficiency seems unlikely, one may wonder if there are specific complexity assumptions that allow to build more efficient digital signature schemes than currently known. Ideally, are there digital signature schemes with $O(k)$ complexity, which can be proved as hard to break as solving a computational problem which is believed to require $2^{\Omega(k)}$ time?

1.1 Results and techniques

The main result in this paper is a construction of a provably secure digital signature scheme with key size and computation time almost linear (up to polylogarithmic factors) in the security parameter. In other words, we give a new digital signature scheme with complexity $O(k \log^c k)$ which can be proved to be as hard to break as a problem which is conjectured to require $2^{\Omega(k)}$ time to solve.

The problem underlying our signature scheme is that of approximating the shortest vector in a lattice with "cyclic" or "ideal" structure, as already used in [22] for the construction of efficient lattice based one-way functions, and subsequently extended to collision resistant functions in [25, 18]. As in most previous work on lattices, our scheme can be proven secure based on the *worst case* complexity of the underlying lattice problems.

Since one-way functions are known to imply the existence of many other cryptographic primitives (e.g., pseudorandom generators, digital signatures, private key encryption, etc.), the efficient lattice based one-way functions of [22] immediately yield corresponding cryptographic primitives based on the complexity of cyclic lattices. However, the known generic constructions of cryptographic primitives from one-way functions are usually very inefficient. So, it was left as an open problem in [22] to find *direct* constructions of other cryptographic primitives from lattice problems with performance and security guarantees similar to those of [22]. For the case of collision resistant hash functions, the problem was resolved in [25, 18], which showed that various variants of the one-way function proposed in [22] are indeed collision resistant. In this paper we build on the results of [22, 25, 18] to build an asymptotically efficient lattice-based digital signature scheme.

Theorem 1. *There exists a signature scheme such that the signature of an n-bit message is of length $\tilde{O}(k)$ and both the signing and verification algorithms take time $\tilde{O}(n)+\tilde{O}(k)$. The scheme is strongly unforgeable in the chosen message attack model, assuming the hardness of approximating the shortest vector problem in all ideal lattices of dimension k to within a factor $\tilde{O}(k^2)$.*

Our lattice based signature scheme is based on a standard transformation from one-time signatures (i.e., signatures that allow to securely sign a single message) to general signature schemes, together with a novel construction of a lattice based one-time signature. We remark that the same transformation from one-time signatures to unrestricted signature schemes was also employed by virtually all previous constructions of digital signatures from arbitrary one-way functions (e.g., [21, 23, 27]). This transformation, which combines one-time signatures together with a tree structure, is relatively efficient and allows one to sign messages with only a logarithmic number of applications of a hash function and a one-time signature scheme [28]. The bottleneck in one-way function based signature schemes is the construction of one-time signatures from one-way functions. The reason for the slowdown is that the one-way function is typically used to sign a k-bit message one bit at a time, so that the entire signature requires k evaluations of the one-way function. In this paper we give a direct construction of one-time signatures, where each signature just requires two applications of the lattice based one-way function of [22, 25, 18]. The same lattice based hash function can then be used to efficiently transform the one-time signature into an unrestricted signature scheme with only a logarithmic loss in performance.

The high level structure of our lattice based one-time signature scheme is easily explained. The construction is based on the generalized compact knapsack functions of [22, 25, 18]. These are keyed functions (indexed by a key $(a_1, \ldots, a_k)$) of the form

$$h(x_1, \ldots, x_m) = \sum_i a_i \cdot x_i,$$

where $a_1, \ldots, a_m, x_1, \ldots, x_m$ are elements of some large ring R, and the result of the function is also in R. The domain of the function is restricted to $x_i \in D$, where D is a subset of R of small elements. For example, if R is the ring of integers, and $D = \{0, 1\}$, then h is just the subset-sum function. Notice that if D is not restricted, then h is certainly not a one-way function: the function can be easily inverted over the integers using the extended Euclid algorithm for greatest common divisor computation. For efficiency reasons, here (as in [22, 25, 18]) we use a different ring R and a much larger subset $D \subset R$, so that a single element of D can be used to encode a k-bit message (see section 2.3). We now give very high level overviews of our one-time signature and the proof of its security.

One-time signature. When the user wants to generate a key for the one-time signature scheme, he simply picks two "random" inputs $\boldsymbol{x}, \boldsymbol{y} \in D^m$, and computes their images under the hash function $(h(\boldsymbol{x}), h(\boldsymbol{y}))$. (The key $(a_1, \ldots, a_m)$ to the hash function h can also be individually chosen by the user, or shared among all the users of the signature scheme.) The secret key is the pair $(\boldsymbol{x}, \boldsymbol{y})$ while the public key is given by their hashes $(h(\boldsymbol{x}), h(\boldsymbol{y}))$. Then, the signature of a message z is simply obtained as a "linear combination" $\boldsymbol{x} \cdot z + \boldsymbol{y}$ of the two secret vectors, with coefficient being the message z to be signed. (The multiplication $\boldsymbol{x} \cdot z$ is defined as the ring multiplication of each coordinate of $\boldsymbol{x}$ by z.) Signatures can be easily verified using the homomorphic properties of the lattice based hash function $h(\boldsymbol{x} \cdot z + \boldsymbol{y}) = h(\boldsymbol{x}) \cdot z + h(\boldsymbol{y})$.

Security proof. If the domain D^m were closed under the ring addition and multiplication operations, then one could show that the public key $(h(\boldsymbol{x}), h(\boldsymbol{y}))$ and signature $\boldsymbol{x} \cdot z + \boldsymbol{y}$ do not reveal enough information to obtain the signer's secret key $(\boldsymbol{x}, \boldsymbol{y})$, and a forgery relative to a *different* secret key will yield a collision to the hash function. But because the domain is restricted, there is a possibility that the signer's secret key was the only one that could have produced $h(\boldsymbol{x}), h(\boldsymbol{y})$ and signature $\boldsymbol{x} \cdot z + \boldsymbol{y}$, and so an adversary who sees these values might be able to deduce the secret key. This turns out to be the main difficulty in carrying out our proof. We overcome this technical problem by choosing the secret key elements $\boldsymbol{x}, \boldsymbol{y}$ according to a carefully crafted (non-uniform) probability distribution, which can be intuitively thought as a "fuzzy" subset of the full domain R^m. It turns out that if the appropriate distribution is used, then we can have the domain D^m be closed under the ring operations in an approximate probabilistic sense, and still have h be a function that's hard to invert.

1.2 Related work

Lamport showed the first construction of a one-time signature based on the existence of one-way functions. In that scheme, the public key consists of the values $f(x_0), f(x_1)$, where f is a one-way function and x_0, x_1 are randomly chosen elements in its domain. The elements x_0 and x_1 are kept secret, and in order to sign a bit i, the signer reveals x_i. This construction requires one application of the one-way function for every bit in the message. Since then, more efficient constructions have been proposed in (e.g. [20, 7, 6, 11, 4, 5, 16]), but there was always an inherent limitation in the number of bits that could be signed efficiently with one application of the one-way function [12].

Provably secure cryptography based on lattice problems was pioneered by Ajtai in [2], and attracted considerable attention within the complexity theory community because of a remarkable worst-case/average-case connection: it is possible to show that breaking the cryptographic function on the average is at least as hard as solving the lattice problem in the worst-case. Unfortunately, functions related to k-dimensional lattices typically involve an k-dimensional matrix/vector multiplication, and therefore require k^2 time to compute (as well as k^2 storage for keys). A fundamental step towards making lattice based cryptography more attractive in practice, was taken by Micciancio [22] who proposed a variant of Ajtai's function which is much more efficient to compute (thanks to the use of certain lattices with a special cyclic structure) and still admits a worst-case/average-case proof of security. The performance improvement in [22] (as well as in subsequent work [25, 18],) comes at a cost: the resulting function is as hard to break as solving the shortest vector problem in the worst case over lattices with a cyclic structure. Still, since the best known algorithms do not perform any better on these lattices than on general ones, it seems reasonable to conjecture that the shortest vector problem is still exponentially hard. It was later shown in [25, 18] that, while the function constructed in [22] was only one-way, it is possible to construct efficient collision-resistant hash functions based on the hardness of problems in lattices with a similar algebraic structure.

1.3 Open problems

Our work raises many interesting open problems. One such problem is constructing a one-time signature with similar efficiency, but based on a weaker hardness assumption. For instance, it would be great to provide a one-time signature with security based on the hardness of approximating the shortest vector problem (in ideal lattices) to within a factor of $\tilde{O}(n)$. Also, with the recent results of Peikert and Rosen [26], showing a possible way to build cryptographic functions whose security is based on approximating the shortest vector in special lattices to within a factor $O(\sqrt{\log n})$, we believe that it is worthwhile exploring whether one-time signatures can be built based on similar assumptions.

Another direction to try to build efficient signature schemes based directly on the hardness of lattice problems without going through one-time signatures and an authentication tree. The main advantage of such a scheme would be that the signer would not have to "keep a state" and remember which verification keys have already been used. Such constructions have been achieved based on problems from number theory [13, 8] but they are not as efficient, in an asymptotic sense, as the signature scheme presented here.

While the scheme presented here has almost optimal asymptotic efficiency, it is not yet ready to be used for practical applications (see Section 4). The main issue is that lattice reduction algorithms perform much better in practice than in theory, and thus our signature scheme may be insecure for parameters appropriate for practical schemes. Nevertheless, the recent advances in lattice-based cryptography are a very encouraging sign that with some novel ideas, our construction can be modified into a serviceable signature scheme.

2 Preliminaries

2.1 Signatures

We recall the definitions of signature schemes and what it means for a signature scheme to be secure.

Definition 2. *A* signature scheme *consists of a triplet of polynomial-time (possibly probabilistic) algorithms* (G, S, V) *such that for every pair of outputs* (s, v) *of* $G(1^n)$ *and any* n*-bit message* m*,*

$$Pr[V(v, m, S(s, m)) = 1] = 1$$

where the probability is taken over the randomness of algorithms S *and* V*.*

In the above definition, G is called the key-generation algorithm, S is the signing algorithm, V is the verification algorithm, and s and v are, respectively, the signing and verification keys.

A signature scheme is said to be secure if there is only a negligible probability that any adversary, after seeing signatures of messages of his choosing, can sign a message whose signature he has not already seen [15]. One-time security means that an adversary, after seeing a signature of a single message of his choosing, cannot produce a valid signature of a different message.

Definition 3. *A signature scheme* (G, S, V) *is said to be* one-time secure *if for every polynomial-time (possibly randomized) adversary* $\mathcal{A}$*, the probability that after seeing* $(m, S(s, m))$ *for any message* m *of its choosing,* $\mathcal{A}$ *can produce* $(m' \neq m, \sigma')$ *such that* $V(v, m', \sigma') = 1$*, is negligibly small. The probability is taken over the randomness of* G*,* S*,* V*, and* $\mathcal{A}$*.*

In the standard security definition of a signature scheme, the adversary should not be able to produce a signature of a message he hasn't already seen. A stronger notion of security, called *strong unforgeability* requires that in addition to the above, an adversary shouldn't even be able to come up with a different signature for a message whose signature he has already seen. The scheme presented in this paper satisfies this stronger notion of unforgeability.

Another feature of signatures that is sometimes desirable is the ability of the legitimate signer to prove that a message was not actually signed by her. Of course, it should be impossible for the signer to repudiate a message that she actually signed. Signatures schemes that have this feature are called Fail-Stop [24]. Our scheme has this property as well.

2.2 Notation

Let $R = \mathbb{Z}_p[x]/\langle f \rangle$ be a ring where f is an irreducible monic polynomial of degree n over $\mathbb{Z}[x]$ and p is some small prime. For the rest of the paper, the variables n, p, and f will always be associated with the ring R. We will denote elements in R by bold letters and elements of R^m, for some positive integer m, by a bold letter with a hat. That is, $\widehat{\boldsymbol{a}} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) \in R^m$ when all the $\boldsymbol{a}_i$'s are in R. For an element $\widehat{\boldsymbol{a}} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) \in R^m$ and an element $\boldsymbol{z} \in R$, we define $\widehat{\boldsymbol{a}}\boldsymbol{z} = (\boldsymbol{a}_1\boldsymbol{z}, \ldots, \boldsymbol{a}_m\boldsymbol{z})$. For two elements $\widehat{\boldsymbol{a}}, \widehat{\boldsymbol{b}} \in R^m$, addition is defined as $\widehat{\boldsymbol{a}}+\widehat{\boldsymbol{b}} = (\boldsymbol{a}_1+\boldsymbol{b}_1, \ldots, \boldsymbol{a}_m+\boldsymbol{b}_m)$ and the dot product as $\widehat{\boldsymbol{a}} \odot \widehat{\boldsymbol{b}} = \boldsymbol{a}_1\boldsymbol{b}_1 + \ldots + \boldsymbol{a}_m\boldsymbol{b}_m$.

Notice that with the operations that we defined, the set R^m is an R-module. That is, R^m is an abelian additive group such that for all $\widehat{\boldsymbol{a}}, \widehat{\boldsymbol{b}} \in R^m$ and $\boldsymbol{r}, \boldsymbol{s} \in R$, we have

1. $(\widehat{\boldsymbol{a}} + \widehat{\boldsymbol{b}})\boldsymbol{r} = \widehat{\boldsymbol{a}}\boldsymbol{r} + \widehat{\boldsymbol{b}}\boldsymbol{r}$
2. $(\widehat{\boldsymbol{a}}\boldsymbol{r})\boldsymbol{s} = \widehat{\boldsymbol{a}}(\boldsymbol{r}\boldsymbol{s})$
3. $\widehat{\boldsymbol{a}}(\boldsymbol{r} + \boldsymbol{s}) = \widehat{\boldsymbol{a}}\boldsymbol{r} + \widehat{\boldsymbol{a}}\boldsymbol{s}$

We will now give a definition for the "length" of elements in R. To do so, we will first need to specify their representations in the ring. For our application, we will represent elements in R by a polynomial of degree $n-1$ having coefficients in the range $[-\frac{p-1}{2}, \frac{p-1}{2}]$, and so when we talk about reduction modulo p, we mean finding an equivalent element modulo p in the aforementioned range. For an element $\boldsymbol{a} = a_0 + a_1x + \ldots + a_{n-1}x^{n-1} \in R$, we define $\|\boldsymbol{a}\|_\infty = max_i(|a_i|)$. Similarly, for elements $\widehat{\boldsymbol{a}} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) \in R^m$, we define $\|\widehat{\boldsymbol{a}}\|_\infty = \max_i (\|\boldsymbol{a}_i\|_\infty)$. Notice that $\|\cdot\|_\infty$ is not exactly a norm because $\|\alpha\boldsymbol{a}\|_\infty \neq \alpha\|\boldsymbol{a}\|_\infty$ for all integers α (because of the reduction modulo p), but it still holds true that $\|\boldsymbol{a}+\boldsymbol{b}\|_\infty \leq \|\boldsymbol{a}\|_\infty + \|\boldsymbol{b}\|_\infty$ and $\|\alpha\boldsymbol{a}\|_\infty \leq \alpha\|\boldsymbol{a}\|_\infty$.

While putting an upper-bound on $\|\boldsymbol{a}+\boldsymbol{b}\|_\infty$ is straight-forward, it turns out that upper-bounding $\|\boldsymbol{a}\boldsymbol{b}\|_\infty$ is somewhat more involved. Suppose that we are trying to determine the upper bound on $\|\boldsymbol{a}\boldsymbol{b}\|_\infty$. For a moment, let's pretend that $\boldsymbol{a}$ and $\boldsymbol{b}$ are polynomials in $\mathbb{Z}[x]$. Then, the product $\boldsymbol{a}\boldsymbol{b}$ will have degree at most $2n-2$ and the absolute value of the maximum coefficient of $\boldsymbol{a}\boldsymbol{b}$ will be at most $n\|\boldsymbol{a}\|_\infty\|\boldsymbol{b}\|_\infty$. Reducing $\boldsymbol{a}\boldsymbol{b}$ modulo p will not increase the absolute value of the maximum coefficient, but reducing modulo the polynomial f can (and usually does). So if we want to upper bound $\|\boldsymbol{a}\boldsymbol{b}\|_\infty$, we need to account for the increase in the coefficient size when we reduce a polynomial in $\mathbb{Z}[x]$ of degree $2n-2$ modulo f.

For any ring R, we define a constant $\phi(R)$ as,

$$\phi(R) = \min\{j : \forall \boldsymbol{a}, \boldsymbol{b} \in R,\ \|\boldsymbol{a}\boldsymbol{b}\|_\infty \leq jn\|\boldsymbol{a}\|_\infty\|\boldsymbol{b}\|_\infty\}.$$

The constant $\phi(R)$ is intimately tied to the concept of "expansion factor" introduced in [18]. It is also somewhat related to the root discriminant of a number field as described in [26]. We will not go into many details here, other than to mention that there are many polynomials f which result in $\phi(R)$ being small and it is not too hard to upper bound the value of $\phi(R)$. For example for $f = x^n + 1$, $\phi(R) = 1$ and for $f = x^n + x^{n-1} + \ldots + 1$, $\phi(R) \leq 2$. In the rest of the paper, we will omit the parameter R, and just write ϕ.

2.3 A hash function family

We now define a function family $\mathcal{H}_{R,m}$ that maps R^m to R. The functions $h \in \mathcal{H}_{R,m}$ are indexed by elements $\widehat{\boldsymbol{a}} \in R^m$. The input to the function is an element $\widehat{\boldsymbol{z}} \in R^m$, and the output is $\widehat{\boldsymbol{a}} \odot \widehat{\boldsymbol{z}}$. So the functions in $\mathcal{H}_{R,m}$ map elements from R^m to R. To summarize,

$$\mathcal{H}_{R,m} = \{h_{\widehat{\boldsymbol{a}}} : \widehat{\boldsymbol{a}} \in R^m\}, \text{ where } h_{\widehat{\boldsymbol{a}}}(\widehat{\boldsymbol{z}}) = \widehat{\boldsymbol{a}} \odot \widehat{\boldsymbol{z}}$$

Throughout the paper, we will write h rather than $h_{\widehat{\boldsymbol{a}}}$ with the understanding that there is an $\widehat{\boldsymbol{a}}$ associated with the function h. Notice that we can efficiently generate random functions from the function family $\mathcal{H}_{R,m}$ by simply generating a random $\widehat{\boldsymbol{a}} \in R^m$.

It was shown in [18] that finding two "small" elements $\widehat{\boldsymbol{s}}, \widehat{\boldsymbol{s}}' \in R^m$ such that $h(\widehat{\boldsymbol{s}}) = h(\widehat{\boldsymbol{s}}')$ for randomly chosen $h \in \mathcal{H}_{R,m}$ is at least as hard as solving the approximate shortest vector problem for *all* lattices of a certain type (a problem which is believed to be hard). The security of our signature scheme will be based on the hardness of this collision problem. We now define the problem formally.

Definition 4. *The collision problem, $Col_{d,\mathcal{H}_{R,m}}(h)$ takes as input a random function $h \in \mathcal{H}_{R,m}$, and asks to find two distinct elements $\widehat{\boldsymbol{s}}, \widehat{\boldsymbol{s}}' \in R^m$ with $\|\widehat{\boldsymbol{s}}\|_\infty, \|\widehat{\boldsymbol{s}}'\|_\infty \leq d$ such that $h(\widehat{\boldsymbol{s}}) = h(\widehat{\boldsymbol{s}}')$.*

We now make some useful observations about the function family $\mathcal{H}_{R,m}$. The first observation is that the functions in $\mathcal{H}_{R,m}$ are module homomorphisms.

Claim. $\mathcal{H}_{R,m}$ is a set of module homomorphisms. That is, for every $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}} \in R^m$, $\boldsymbol{z} \in R$, and $h \in \mathcal{H}_{R,m}$, the following two conditions are satisfied:

1. $h(\widehat{\boldsymbol{k}} + \widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{k}}) + h(\widehat{\boldsymbol{l}})$
2. $h(\widehat{\boldsymbol{k}}\boldsymbol{z}) = h(\widehat{\boldsymbol{k}})\boldsymbol{z}$

Proof. By the definition of the hash function h, we have

1. $h(\widehat{\boldsymbol{k}} + \widehat{\boldsymbol{l}}) = \widehat{\boldsymbol{a}} \odot (\widehat{\boldsymbol{k}} + \widehat{\boldsymbol{l}}) = \widehat{\boldsymbol{a}} \odot \widehat{\boldsymbol{k}} + \widehat{\boldsymbol{a}} \odot \widehat{\boldsymbol{l}} = h(\widehat{\boldsymbol{k}}) + h(\widehat{\boldsymbol{l}})$
2. $h(\widehat{\boldsymbol{k}}\boldsymbol{z}) = \widehat{\boldsymbol{a}} \odot (\boldsymbol{k}_1\boldsymbol{z}, \ldots, \boldsymbol{k}_m\boldsymbol{z}) = \boldsymbol{a}_1\boldsymbol{k}_1\boldsymbol{z} + \ldots + \boldsymbol{a}_m\boldsymbol{k}_m\boldsymbol{z}$
$= (\boldsymbol{a}_1\boldsymbol{k}_1 + \ldots + \boldsymbol{a}_m\boldsymbol{k}_m)\boldsymbol{z} = (\widehat{\boldsymbol{a}} \odot \widehat{\boldsymbol{k}})\boldsymbol{z} = h(\widehat{\boldsymbol{k}})\boldsymbol{z}$

□

The next observation is that the kernel of every $h \in \mathcal{H}_{R,m}$ contains an exponential number of "small" elements.

Lemma 5. *For every $h \in \mathcal{H}_{R,m}$, there exist at least 5^{mn} elements $\widehat{\boldsymbol{y}} \in R^m$ such that $\|\widehat{\boldsymbol{y}}\|_\infty \leq 5p^{1/m}$ and $h(\widehat{\boldsymbol{y}}) = \boldsymbol{0}$.*

Proof. Let S be the set containing all elements in R^m with coefficients between 0 and $5p^{1/m}$. Since $|S| = (5p^{1/m} + 1)^{mn} > 5^{mn}p^n$ and $|R| = p^n$, by the pigeonhole principle, there exists a $\boldsymbol{t} \in R$ and a subset $S' \subseteq S$ such that $|S'| \geq 5^{mn}$ and for all $\widehat{\boldsymbol{s}}' \in S'$, $h(\widehat{\boldsymbol{s}}') = \boldsymbol{t}$. If $\boldsymbol{t} = \boldsymbol{0}$, then we're done, otherwise let $S' = \{\widehat{\boldsymbol{s}}'_1, \widehat{\boldsymbol{s}}'_2, \ldots, \widehat{\boldsymbol{s}}'_k\}$ and consider the set $Y = \{\widehat{\boldsymbol{s}}'_1 - \widehat{\boldsymbol{s}}'_1, \widehat{\boldsymbol{s}}'_1 - \widehat{\boldsymbol{s}}'_2, \ldots, \widehat{\boldsymbol{s}}'_1 - \widehat{\boldsymbol{s}}'_k\}$ of size $|S'|$. Note that for each $\widehat{\boldsymbol{y}} \in Y$, $\|\widehat{\boldsymbol{y}}\|_\infty \leq 5p^{1/m}$ and $h(\widehat{\boldsymbol{y}}) = \boldsymbol{0}$ because of the homomorphic property of h. □

2.4 Lattices

In this subsection, we explain the relationship between the collision problem from Definition 4 and finding shortest vectors in certain types of lattices.

An n-dimensional integer lattice $\mathcal{L}$ is a subgroup of $\mathbb{Z}^n$. A lattice $\mathcal{L}$ can be represented by a set of linearly independent generating vectors, called a basis. For any lattice vector $\boldsymbol{y} \in \mathcal{L}$, the infinity norm of $\boldsymbol{y}$, $\|\boldsymbol{y}\|_\infty$, is the absolute value of the largest coefficient of $\boldsymbol{y}$. The *minimum distance* (in the infinity norm[1]) of a lattice $\mathcal{L}$, denoted by $\lambda_1(\mathcal{L})$, is defined as:

$$\lambda_1(\mathcal{L}) = \min_{\boldsymbol{y} \in \mathcal{L} \setminus \{\boldsymbol{0}\}} \{\|\boldsymbol{y}\|_\infty\}$$

Computing the $\lambda_1(\mathcal{L})$ of a lattice was first shown to be NP-hard by van Emde Boas [29], and it was shown hard to approximate to within a factor of $n^{1/\log \log n}$ by Dinur [10]. It is conjectured that approximating $\lambda_1(\mathcal{L})$ to within any polynomial factor is a hard problem (though not NP-hard [14, 1]) since the fastest known algorithm takes time $2^{O(n)}$ to accomplish this [17, 3].

Micciancio [22] defined a cyclic lattice to be a lattice $\mathcal{L}$ such that if the vector $(a_1, \ldots, a_{n-1}, a_n) \in \mathcal{L}$, then the vector $(a_n, a_1, \ldots, a_{n-1})$ is also in the lattice $\mathcal{L}$. Such lattices correspond to ideals in $\mathbb{Z}[x]/\langle x^n - 1\rangle$. In [22], Micciancio gave a construction of an efficient family of one-way functions with security based on the worst case hardness of approximating $\lambda_1(\mathcal{L})$ in cyclic lattices. Subsequently, it was shown in [25, 18] how to modify Micciancio's function in order to make it collision resistant. In addition, it was shown in [18] how to create efficient collision resistant hash functions with security based on approximating $\lambda_1(\mathcal{L})$ in lattices that correspond to ideals in rings $\mathbb{Z}[x]/\langle f\rangle$ for general f. A lattice corresponding to an ideal means that the vector $(a_0, \ldots, a_{n-1})$ is in the lattice, if and only if the polynomial $a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$ is in the ideal. Despite the added structure of these algebraic lattices, the best algorithms to solve the shortest vector problem are the same ones as for arbitrary lattices.

The following theorem is a weaker special case of the main result of [18] that is most pertinent to this work:

Theorem 6. *Let f be an irreducible polynomial in $\mathbb{Z}[x]$ of degree n and define integers $p = (\phi n)^3$ and $m = \lceil \log n \rceil$. If there exists a polynomial-time algorithm that solves $Col_{d,\mathcal{H}_{R,m}}(h)$ for $R = \mathbb{Z}_p[x]/\langle f\rangle$ and $d = 10\phi p^{1/m} n \log^2 n$, then there exists a polynomial-time algorithm that approximates $\lambda_1(\mathcal{L})$ to within a factor of $\tilde{O}(\phi^5 n^2)$ for every lattice $\mathcal{L}$ corresponding to an ideal in the ring $\mathbb{Z}[x]/\langle f\rangle$.*

We point out that in [18, Theorem 2] (which is the main result of [18]), it is shown that solving the $Col_{d,\mathcal{H}_{R,m}}(h)$ problem for certain parameters p, d, and m implies approximating the shortest vector to within a factor of $\tilde{O}(n)$. Unfortunately, in the current paper we cannot show that breaking the one-time

[1] All the results in this paper can be adapted to any ℓ_p norm. For simplicity, we concentrate on the ℓ_∞ case, since it is the most convenient one in cryptographic applications

signature implies solving the $Col_{d,\mathcal{H}_{R,m}}(h)$ problem for such optimal parameters (mainly, we cannot get the parameter d to be too small). And so Theorem 6 is a weaker version of [18, Theorem 2] where the parameters p, m, and d are set in a way such that breaking the one-time signature implies solving $Col_{d,\mathcal{H}_{R,m}}(h)$.

We also notice in the above theorem that the approximation factor heavily depends on ϕ. Thus it's prudent to choose a polynomial f that results in a small ϕ. Choosing irreducible polynomials of the form $x^n + 1$ or $x^n + x^{n-1} + \ldots + 1$ makes ϕ a small constant (1 and 2 respectively). We also point out that the integer p needs not be a prime for the proof of security to hold, but there are some practical advantages to setting it to a prime when implementing functions that involve multiplications of elements in $\mathbb{Z}_p[x]/\langle f \rangle$ [19].

3 The One-Time Signature Scheme

In this section we present our one-time signature scheme. The security of the scheme will be ultimately based on the worst-case hardness of approximating the shortest vector in all lattices corresponding to ideals in the ring $\mathbb{Z}[x]/\langle f \rangle$ for any irreducible polynomial f. The approximation factor is determined by the polynomial f as in Theorem 6. The key-generation algorithm for the signature scheme allows us to specify the polynomial f that we want to use for the hardness assumption.

Key-Generation Algorithm:
Input: 1^n, irreducible polynomial $f \in \mathbb{Z}[x]$ of degree n.

Set $p \leftarrow (\phi n)^3$, $m \leftarrow \lceil \log n \rceil$, $R \leftarrow \mathbb{Z}_p[x]/\langle f \rangle$
For all positive i, let the sets DK_i and DL_i be defined as:

$$DK_i = \{\widehat{\boldsymbol{y}} \in R^m \text{ such that } \|\widehat{\boldsymbol{y}}\|_\infty \leq 5ip^{1/m}\}$$

$$DL_i = \{\widehat{\boldsymbol{y}} \in R^m \text{ such that } \|\widehat{\boldsymbol{y}}\|_\infty \leq 5in\phi p^{1/m}\}$$

Choose uniformly random $h \in \mathcal{H}_{R,m}$
Pick a uniformly random string $r \in \{0,1\}^{\lfloor \log^2 n \rfloor}$
if $r = 0^{\lfloor \log^2 n \rfloor}$ **then**
    set $j = \lfloor \log^2 n \rfloor$
else
    set j to the position of the first 1 in the string r
end if
Pick $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$ independently and uniformly at random from DK_j and DL_j respectively
Signing Key: $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$. Verification Key: $(h, h(\widehat{\boldsymbol{k}}), h(\widehat{\boldsymbol{l}}))$

Signing Algorithm:
Input: Message $\boldsymbol{z} \in R$ such that $\|\boldsymbol{z}\|_\infty \leq 1$; signing key $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$
Output: $\widehat{\boldsymbol{s}} \leftarrow \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$

Verification Algorithm:
Input: Message $\boldsymbol{z}$; signature $\widehat{\boldsymbol{s}}$; verification key $(h, h(\widehat{\boldsymbol{k}}), h(\widehat{\boldsymbol{l}}))$
Output: "ACCEPT", if $\|\widehat{\boldsymbol{s}}\|_\infty \leq 10\phi p^{1/m} n \log^2 n$ and $h(\widehat{\boldsymbol{s}}) = h(\widehat{\boldsymbol{k}})\boldsymbol{z} + h(\widehat{\boldsymbol{l}})$
"REJECT", otherwise.

At this point we would like to draw the reader's attention to the particulars of how the key-generation algorithm generates the secret signing key $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$. Because of the way that the integer j is generated, the secret key $\widehat{\boldsymbol{k}}$ (resp. $\widehat{\boldsymbol{l}}$) gets chosen uniformly at random from the set DK_j (resp. DL_j) with probability 2^{-j} for $1 \leq j < \lfloor \log^2 n \rfloor$ and with probability 2^{-j+1} for $j = \lfloor \log^2 n \rfloor$. Since $DK_1 \subset DK_2 \subset \ldots \subset DK_{\lfloor \log^2 n \rfloor}$ and $DL_1 \subset DL_2 \subset \ldots \subset DL_{\lfloor \log^2 n \rfloor}$, the keys $\widehat{\boldsymbol{k}}$ and $\widehat{\boldsymbol{l}}$ end up being chosen from the sets $DK_{\lfloor \log^2 n \rfloor}$ and $DL_{\lfloor \log^2 n \rfloor}$, but *not* uniformly at random. Notice that keys with smaller coefficients are more likely to be chosen, and it's also extremely unlikely that we will ever end up with keys that are not in $DK_{\lfloor \log^2 n \rfloor - 1}$ and $DL_{\lfloor \log^2 n \rfloor - 1}$. So with probability negligibly close to 1, there will always be valid secret keys that are "larger" than the ones generated by the key-generation algorithm. This will be crucial to the proof of security.

We will first show that the verification algorithm will always accept the signature generated by the signing algorithm of any message $\boldsymbol{z} \in R$. Note that the signing keys $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$ are contained in sets $DK_{\log^2 n}$ and $DL_{\log^2 n}$ respectively. Thus $\|\widehat{\boldsymbol{k}}\|_\infty \leq 5p^{1/m} \log^2 n$ and $\|\widehat{\boldsymbol{l}}\|_\infty \leq 5\phi p^{1/m} n \log^2 n$. Therefore,

$$\|\widehat{\boldsymbol{s}}\|_\infty = \|\widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}\|_\infty \leq \|\widehat{\boldsymbol{k}}\boldsymbol{z}\|_\infty + \|\widehat{\boldsymbol{l}}\|_\infty \leq \phi n \|\widehat{\boldsymbol{k}}\|_\infty \|\boldsymbol{z}\|_\infty + \|\widehat{\boldsymbol{l}}\|_\infty \leq 10\phi p^{1/m} n \log^2 n$$

Also, by the homomorphic property of functions $h \in \mathcal{H}_{R,m}$,

$$h(\widehat{\boldsymbol{s}}) = h(\widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{k}})\boldsymbol{z} + h(\widehat{\boldsymbol{l}}).$$

We next show that the above signature scheme is secure against forgery. More precisely, we show that forging a signature implies being able to solve the $Col_{d,\mathcal{H}_{R,m}}(h)$ problem for the parameters in Theorem 6, which in turn implies being able to approximate $\lambda_1(\mathcal{L})$ for any lattice $\mathcal{L}$ that corresponds to an ideal in the ring $\mathbb{Z}[x]/\langle f \rangle$.

Theorem 7. *If there exists a polynomial-time adversary that, after seeing a signature $\widehat{\boldsymbol{s}} = \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$ of a message $\boldsymbol{z}$, can output a valid signature of another message $\boldsymbol{z}'$ with probability $1/poly(n)$, then there exists a polynomial time algorithm that can solve the $Col_{d,\mathcal{H}_{R,m}}(h)$ problem for $d = 10\phi p^{1/m} n \log^2 n$.*

Proof. Let $\mathcal{A}$ be an adversary who can break the one-time signature scheme. This means that after seeing a signature for any message of his choice, $\mathcal{A}$ can then successfully sign a different message of his choice.

Before proceeding any further, we point out that an adversary who succeeds in forging a signature with non-negligible probability must succeed with non-negligible probability in the case that $j < \lfloor \log^2 n \rfloor$ in the key-generation step.

This is because j equals $\lfloor \log^2 n \rfloor$ with probability only $2^{-\lfloor \log^2 n \rfloor + 1}$, and so an adversary must also be able to forge signatures for other values of j if he is to have a non-negligible success probability. In the remainder of the proof, we will be assuming that the j generated in the key-generation step was less than $\lfloor \log^2 n \rfloor$. In other words, we'll be assuming that $\widehat{\boldsymbol{k}} \in DK_{\lfloor \log^2 n - 1 \rfloor}$ and $\widehat{\boldsymbol{l}} \in DL_{\lfloor \log^2 n - 1 \rfloor}$.

The algorithm below uses the message-forging adversary $\mathcal{A}$ to solve the $Col_{d,\mathcal{H}_{R,m}}(h)$ problem for the parameters specified in Theorem 6.

$Col_{d,\mathcal{H}_{R,m}}(h)$
1: Run the Key-Generation algorithm (but use the given h instead of generating a random one).
2: Receive message $\boldsymbol{z}$ from $\mathcal{A}$.
3: Send $\widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$ to $\mathcal{A}$.
4: Receive message $\boldsymbol{z}'$ and its signature $\widehat{\boldsymbol{s}}'$ from $\mathcal{A}$
5: Output $\widehat{\boldsymbol{s}}'$ and $\widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$

We now need to show that the outputs of the above algorithm are a collision for the function h with non-negligible probability. If $\mathcal{A}$ succeeds in forging a signature $\widehat{\boldsymbol{s}}'$ for $\boldsymbol{z}'$ (which happens with non-negligible probability), then $\|\widehat{\boldsymbol{s}}'\|_\infty \leq 10\phi p^{1/m} n \log^2 n$ and $h(\widehat{\boldsymbol{s}}') = h(\widehat{\boldsymbol{k}})\boldsymbol{z}' + h(\widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}})$. So if $\widehat{\boldsymbol{s}}' \neq \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$, then our algorithm outputted two distinct elements that form a collision for the function h.

On the other hand, if $\widehat{\boldsymbol{s}}' = \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$, then we do not get a collision. To complete the proof of Theorem 7, we will show that it's extremely unlikely that an adversary (even one with unlimited computational power) can produce an $\widehat{\boldsymbol{s}}'$ and a $\boldsymbol{z}'$ such that $\widehat{\boldsymbol{s}}' = \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$. This will be done in two steps. In the first step, we show that being able to produce such an $\widehat{\boldsymbol{s}}'$ and $\boldsymbol{z}'$ implies uniquely determining the signing key $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$. Then in the second step we show that given the public key $(h, h(\widehat{\boldsymbol{k}}), h(\widehat{\boldsymbol{l}}))$ and a signature $\widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$ of message $\boldsymbol{z}$, it is *information theoretically* impossible to determine the signing key $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$. This means that if $\mathcal{A}$ is able to forge a signature $\widehat{\boldsymbol{s}}'$ for some message $\boldsymbol{z}'$, then almost certainly $\widehat{\boldsymbol{s}}' \neq \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$.

We now show that obtaining an $\widehat{\boldsymbol{s}}'$ and a $\boldsymbol{z}'$ such that $\widehat{\boldsymbol{s}}' = \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$ uniquely determines $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$. Since we know that $\widehat{\boldsymbol{s}} = \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$ and $\widehat{\boldsymbol{s}}' = \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$, it follows that $\widehat{\boldsymbol{s}} - \widehat{\boldsymbol{s}}' = \widehat{\boldsymbol{k}}(\boldsymbol{z} - \boldsymbol{z}')$. Since $\|\widehat{\boldsymbol{k}}\|_\infty \leq 5p^{1/m} \log^2 n$ and $\|\boldsymbol{z} - \boldsymbol{z}'\|_\infty \leq 2$, multiplying $\widehat{\boldsymbol{k}}$ by $\boldsymbol{z} - \boldsymbol{z}'$ in the ring $\mathbb{Z}_p[x]/\langle f \rangle$ is the same as multiplying them in the ring $\mathbb{Z}[x]/\langle f \rangle$ because the coefficients never get big enough to get reduced modulo p. This is because

$$\|\widehat{\boldsymbol{k}}(\boldsymbol{z} - \boldsymbol{z}')\|_\infty \leq 10\phi p^{1/m} n \log^2 n = 80\phi^{1+\frac{3}{\log n}} n \log^2 n = \phi^{1+o(1)} \cdot o(n^2),$$

but in order to get reduced modulo p, the absolute value of the coefficients would have to be at least $p/2 = \Theta(\phi^3 n^3)$, which is a much larger quantity. Now, since the ring $\mathbb{Z}[x]/\langle f \rangle$ is an integral domain and $\boldsymbol{z} - \boldsymbol{z}' \neq \boldsymbol{0}$, there cannot exist another key $\widehat{\boldsymbol{k}}' \neq \widehat{\boldsymbol{k}}$ such that $\widehat{\boldsymbol{k}}'(\boldsymbol{z} - \boldsymbol{z}') = \widehat{\boldsymbol{k}}(\boldsymbol{z} - \boldsymbol{z}')$. And so the key $\widehat{\boldsymbol{k}}$ is uniquely determined (and is equal to $\frac{\widehat{\boldsymbol{s}} - \widehat{\boldsymbol{s}}'}{\boldsymbol{z} - \boldsymbol{z}'}$), and similarly the key $\widehat{\boldsymbol{l}} = \widehat{\boldsymbol{s}} - \widehat{\boldsymbol{k}}\boldsymbol{z}$ is also unique.

Now we move on to showing that by knowing only $h, h(\widehat{\boldsymbol{k}}), h(\widehat{\boldsymbol{l}}), \boldsymbol{z}$, and $\widehat{\boldsymbol{k}}\boldsymbol{z}+\widehat{\boldsymbol{l}}$, it is information theoretically impossible to determine the signing key $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$ (and thus, information theoretically impossible to come up with $\widehat{\boldsymbol{s}}', \boldsymbol{z}'$ such that $\widehat{\boldsymbol{s}}' = \widehat{\boldsymbol{k}}\boldsymbol{z}' + \widehat{\boldsymbol{l}}$). The idea is to show that for every $h, h(\widehat{\boldsymbol{k}}), h(\widehat{\boldsymbol{l}}), \boldsymbol{z}, \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$ there is an exponential number of signing keys $(\widehat{\boldsymbol{k}}', \widehat{\boldsymbol{l}}')$, other than $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$, that satisfy $h(\widehat{\boldsymbol{k}}) = h(\widehat{\boldsymbol{k}}')$, $h(\widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{l}}')$, and $\widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}} = \widehat{\boldsymbol{k}}'\boldsymbol{z} + \widehat{\boldsymbol{l}}'$. And in addition, the total probability that one of these other keys was chosen in the key-generation step (conditioned on $h, h(\widehat{\boldsymbol{k}}), h(\widehat{\boldsymbol{l}}), \boldsymbol{z}, \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$) is almost one.

We point out that we are not proving *witness-indistinguishability*. It's actually quite possible that for every other key $(\widehat{\boldsymbol{k}}', \widehat{\boldsymbol{l}}')$, the probability that it was the key that was used to sign the message is exponentially smaller than the probability that $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$ was the key. What we will be showing is that the sum of probabilities of all other possible keys *combined* being the secret key is exponentially *larger* than the probability that $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$ was the key.

Lemma 8. *Let $(h, \boldsymbol{K}, \boldsymbol{L})$ be the verification key of the signature scheme and $\widehat{\boldsymbol{s}}$ is the signature of some message $\boldsymbol{z}$. Then for any signing key $(\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})$ such that $\widehat{\boldsymbol{k}} \in DK_{\lfloor \log^2 n - 1 \rfloor}, \widehat{\boldsymbol{l}} \in DL_{\lfloor \log^2 n - 1 \rfloor}$, $h(\widehat{\boldsymbol{k}}) = \boldsymbol{K}$, $h(\widehat{\boldsymbol{l}}) = \boldsymbol{L}$ and $\widehat{\boldsymbol{s}} = \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$, the probability that this was the actual signing key generated by the key-generation algorithm is negligibly small.*

Proof. We define the set Y to be the elements of the kernel of h that have "small lengths". In particular,

$$Y = \{\widehat{\boldsymbol{y}} \in R^m \text{ such that } \|\widehat{\boldsymbol{y}}\|_\infty \leq 5p^{1/m} \text{ and } h(\widehat{\boldsymbol{y}}) = \boldsymbol{0}\}.$$

For every $\widehat{\boldsymbol{y}} \in Y$, consider the elements $\widehat{\boldsymbol{k}}' = \widehat{\boldsymbol{k}} - \widehat{\boldsymbol{y}}$ and $\widehat{\boldsymbol{l}}' = \widehat{\boldsymbol{l}} + \widehat{\boldsymbol{y}}\boldsymbol{z}$. Notice that

$$h(\widehat{\boldsymbol{k}}') = h(\widehat{\boldsymbol{k}} - \widehat{\boldsymbol{y}}) = h(\widehat{\boldsymbol{k}}) - h(\widehat{\boldsymbol{y}}) = \boldsymbol{K} - \boldsymbol{0} = \boldsymbol{K},$$

$$h(\widehat{\boldsymbol{l}}') = h(\widehat{\boldsymbol{l}} + \widehat{\boldsymbol{y}}\boldsymbol{z}) = h(\widehat{\boldsymbol{l}}) + h(\widehat{\boldsymbol{y}})\boldsymbol{z} = \boldsymbol{L} + \boldsymbol{0} = \boldsymbol{L},$$

$$\widehat{\boldsymbol{k}}'\boldsymbol{z} + \widehat{\boldsymbol{l}}' = (\widehat{\boldsymbol{k}} - \widehat{\boldsymbol{y}})\boldsymbol{z} + \widehat{\boldsymbol{l}} + \widehat{\boldsymbol{y}}\boldsymbol{z} = \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}} = \widehat{\boldsymbol{s}}.$$

Thus, for every $\widehat{\boldsymbol{y}} \in Y$, if $\widehat{\boldsymbol{k}}'$ happens to be in $DK_{\lfloor \log^2 n \rfloor}$ and $\widehat{\boldsymbol{l}}'$ happens to be in $DL_{\lfloor \log^2 n \rfloor}$, then $(\widehat{\boldsymbol{k}}', \widehat{\boldsymbol{l}}')$ is another valid signing key that could have been used to sign the message $\boldsymbol{z}$. Since $\|\widehat{\boldsymbol{y}}\|_\infty \leq 5p^{1/m}$ and $\|\widehat{\boldsymbol{y}}\boldsymbol{z}\|_\infty \leq 5n\phi p^{1/m}$, we get the following bounds on the norms of $\widehat{\boldsymbol{k}}'$ and $\widehat{\boldsymbol{l}}'$:

$$\|\widehat{\boldsymbol{k}}'\|_\infty \leq \|\widehat{\boldsymbol{k}}\|_\infty + \|\widehat{\boldsymbol{y}}\|_\infty \leq \|\widehat{\boldsymbol{k}}\|_\infty + 5p^{1/m},$$

$$\|\widehat{\boldsymbol{l}}'\|_\infty \leq \|\widehat{\boldsymbol{l}}\|_\infty + \|\widehat{\boldsymbol{y}}\boldsymbol{z}\|_\infty \leq \|\widehat{\boldsymbol{l}}\|_\infty + 5n\phi p^{1/m}.$$

For the remainder of the proof, let i be the smallest integer such that $\widehat{\boldsymbol{k}}$ and $\widehat{\boldsymbol{l}}$ are contained in DK_i and DL_i respectively. Then $\widehat{\boldsymbol{k}}'$ and $\widehat{\boldsymbol{l}}'$ are definitely contained in DK_{i+1} and DL_{i+1} for every $\widehat{\boldsymbol{y}} \in Y$. And since we assumed that $\widehat{\boldsymbol{k}} \in DK_{\lfloor \log^2 n-1 \rfloor}$ and $\widehat{\boldsymbol{l}} \in DL_{\lfloor \log^2 n-1 \rfloor}$, it turns out that $(\widehat{\boldsymbol{k}}', \widehat{\boldsymbol{l}}')$ is a perfectly valid signing key. To prove the lemma, we will need to upper-bound the probability that the generated secret keys were $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$ given that the public keys are $\boldsymbol{K} = h(\widehat{\boldsymbol{k}})$ and $\boldsymbol{L} = h(\widehat{\boldsymbol{l}})$ and the signature of $\boldsymbol{z}$ is $\widehat{\boldsymbol{s}} = \widehat{\boldsymbol{k}}\boldsymbol{z} + \widehat{\boldsymbol{l}}$. Let E be the event that the verification key are $\boldsymbol{K}$ and $\boldsymbol{L}$ and the signature of $\boldsymbol{z}$ is $\widehat{\boldsymbol{s}}$.

$$Pr[\text{signing key} = (\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})|E] = \frac{Pr[\text{key} = (\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})\,\&\,E]}{Pr[E]} = \frac{Pr[\text{key} = (\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})]}{Pr[E]}$$

We now calculate the probability that the keys were $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$. This is computed by noting that $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$ were generated by selecting $j \geq i$ with probability 2^{-j} and then selecting $\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}}$ from DK_j and DL_j. Since $\widehat{\boldsymbol{k}}$ and $\widehat{\boldsymbol{l}}$ are chosen uniformly and independently at random from DK_j and DL_j, the probability that they are both chosen is $\frac{1}{|DK_j| \cdot |DL_j|}$. So,

$$Pr[\text{signing key} = (\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})] = \frac{1}{2^i|DK_i||DL_i|} + \frac{1}{2^{i+1}|DK_{i+1}||DL_{i+1}|} + \dots \tag{1}$$

To calculate the probability of event E, we need to figure out the probability that the keys chosen will result in public keys $\boldsymbol{K}$ and $\boldsymbol{L}$ and when given the message $\boldsymbol{z}$, the signature will be $\widehat{\boldsymbol{s}}$. We have shown above that for every $\widehat{\boldsymbol{y}} \in Y$, choosing the keys $\widehat{\boldsymbol{k}} - \widehat{\boldsymbol{y}}, \widehat{\boldsymbol{l}} + \widehat{\boldsymbol{y}}\boldsymbol{z}$ will produce public keys $\boldsymbol{K}, \boldsymbol{L}$ and signature $\widehat{\boldsymbol{s}}$. Since we know that $\widehat{\boldsymbol{k}} - \widehat{\boldsymbol{y}}$ and $\widehat{\boldsymbol{l}} + \widehat{\boldsymbol{y}}\boldsymbol{z}$ are contained in DK_{i+1} and DL_{i+1} respectively, we get

$$Pr[E] > \frac{|Y|}{2^{i+1}|DK_{i+1}||DL_{i+1}|} + \frac{|Y|}{2^{i+2}|DK_{i+2}||DL_{i+2}|} + \dots \tag{2}$$

If we let $q = Pr[\text{signing key} = (\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})]$, then combining (1) and (2) we get

$$Pr[E] > |Y| \left(q - \frac{1}{2^i|DK_i||DL_i|} \right)$$

and so,

$$\begin{aligned}\frac{Pr[\text{signing key} = (\widehat{\boldsymbol{k}}, \widehat{\boldsymbol{l}})]}{Pr[E]} &< \frac{q}{|Y|\left(q - \frac{1}{2^i|DK_i||DL_i|}\right)} = \frac{q2^i|DK_i||DL_i|}{|Y|(q2^i|DK_i||DL_i| - 1)} \\ &= \frac{1}{|Y|}\left(1 + \frac{1}{q2^i|DK_i||DL_i| - 1}\right)\end{aligned}$$

Before proceeding, we will state the following inequality that will be used later,

$$\begin{aligned}\frac{|DK_{i+1}||DL_{i+1}|}{|DK_i||DL_i|} &= \frac{(2 \cdot 5(i+1)p^{1/m})^{mn}(2 \cdot 5(i+1)n\phi p^{1/m})^{mn}}{(2 \cdot 5ip^{1/m})^{mn}(2 \cdot 5in\phi p^{1/m})^{mn}} \\ &= \left(1 + \frac{1}{i}\right)^{2mn} \leq 2^{2mn} = 4^{mn}\end{aligned}$$

Now we use the above inequality to lower bound the quantity $q2^i|DK_i||DL_i|$. Recall that q was defined to be the probability that the signing key is $(\widehat{\boldsymbol{k}},\widehat{\boldsymbol{l}})$, and so from Equation (1), we obtain

$$\begin{aligned} q2^i|DK_i||DL_i| &= 2^i|DK_i||DL_i|\left(\frac{1}{2^i|DK_i||DL_i|}+\frac{1}{2^{i+1}|DK_{i+1}||DL_{i+1}|}+\ldots\right) \\ &> 2^i|DK_i||DL_i|\left(\frac{1}{2^i|DK_i||DL_i|}+\frac{1}{2^{i+1}|DK_{i+1}||DL_{i+1}|}\right) \\ &= 1+\frac{|DK_i||DL_i|}{2|DK_{i+1}||DL_{i+1}|} \geq 1+\frac{1}{2\cdot 4^{mn}} \end{aligned}$$

Using the above inequality, we obtain

$$\frac{Pr[\text{signing key} = (\widehat{\boldsymbol{k}},\widehat{\boldsymbol{l}})]}{Pr[E]} < \frac{1}{|Y|}\left(1+\frac{1}{q2^i|DK_i||DL_i|-1}\right) \leq \frac{1}{|Y|}(1+2\cdot 4^{mn})$$

and since by Lemma 5 we know that $|Y| \geq 5^{mn}$, we are done. □

This concludes the proof of the theorem. □

3.1 Strong unforgeability

We now show that our one-time signature scheme also satisfies a stronger notion of security, called *strong unforgeability*. In the previous section we showed that if an adversary can produce a signature for an unseen message, then $Col_{d,\mathcal{H}_{R,m}}(h)$ can be solved in polynomial time. Now we point out that $Col_{d,\mathcal{H}_{R,m}}(h)$ can be solved in polynomial time even if the adversary is able to produce a different signature of a message whose signature he has seen. Suppose that after seeing the signature $\widehat{\boldsymbol{s}} = \widehat{\boldsymbol{k}}\boldsymbol{z}+\widehat{\boldsymbol{l}}$ of a message $\boldsymbol{z}$, the adversary $\mathcal{A}$ sends back another valid signature $\widehat{\boldsymbol{s}}' \neq \widehat{\boldsymbol{s}}$ of $\boldsymbol{z}$. Then $\widehat{\boldsymbol{s}}$ and $\widehat{\boldsymbol{s}}'$ form a collision for h. This is because

$$h(\widehat{\boldsymbol{s}}') = h(\widehat{\boldsymbol{k}})\boldsymbol{z}+h(\widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{k}}\boldsymbol{z}+\widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{s}}).$$

4 Practical attacks

While our scheme is efficient and secure in an asymptotic sense, it is not yet secure for parameters that one would want to use in practical applications. In this section we demonstrate an attack against our one-time signature scheme by showing how an adversary would go about forging a signature for the message $\boldsymbol{z} = \boldsymbol{0}$. We demonstrate the attack for this message because it is the simplest to explain, but the attack can be easily adapted to any other message.

Knowing the public keys $h(\widehat{\boldsymbol{k}})$ and $h(\widehat{\boldsymbol{l}})$, we can forge a signature for the message $\boldsymbol{z} = \boldsymbol{0}$ by finding an element $\widehat{\boldsymbol{l}}'$ of length less than $10\phi p^{1/m} n \log^2 n$ such that $h(\widehat{\boldsymbol{l}}') = h(\widehat{\boldsymbol{l}})$ and outputting it as the signature $\widehat{\boldsymbol{s}}$. Note that $h(\widehat{\boldsymbol{s}}) = h(\widehat{\boldsymbol{k}})\boldsymbol{0}+h(\widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{l}}) = h(\widehat{\boldsymbol{l}}')$ and also $\|\widehat{\boldsymbol{s}}\|_\infty = \|\widehat{\boldsymbol{l}}'\|_\infty$. So $\widehat{\boldsymbol{s}}$ will be a valid signature of $\mathbf{0}$. The hard part is of course finding an $\widehat{\boldsymbol{l}}'$ such that $h(\widehat{\boldsymbol{l}}') = h(\widehat{\boldsymbol{l}})$. But while this problem is believed to be exponentially hard in n (the degree of the polynomial $h(\widehat{\boldsymbol{l}})$),for small values of n, this problem is heuristically solvable. We will now give an overview of how one would go about finding an $\widehat{\boldsymbol{l}}'$ with small coefficients when given $h(\widehat{\boldsymbol{l}}')$.

The idea is to use lattice reduction and so first we will need to view multiplication in the ring $\mathbb{Z}_p[x]/\langle f\rangle$ as matrix-vector multiplication. Every polynomial in $\mathbb{Z}_p[x]/\langle f\rangle$ can be associated with an n-dimensional vector in $\mathbb{Z}_p$ in the obvious way. Also, for any element $\boldsymbol{a} \in \mathbb{Z}_p[x]/\langle f\rangle$, define $M(\boldsymbol{a})$ to be an $n \times n$ matrix where the i^{th} column (for $0 \leq i \leq n-1$) corresponds to the vector representation of the polynomial $\boldsymbol{a}\boldsymbol{x}^i$. Now we can see that the multiplication of two polynomials $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}_p[x]/\langle f\rangle$ can be written as the multiplication modulo p of the matrix $M(\boldsymbol{a})$ by the vector representation of $\boldsymbol{b}$.

By the above observation, the evaluation of the function $h_{\widehat{\boldsymbol{a}}}(\widehat{\boldsymbol{l}}')$ can be interpreted as as a multiplication of an $n \times nm$ matrix $\boldsymbol{A} = (M(\boldsymbol{a}_1)|\ldots|M(\boldsymbol{a}_m))$ by the vector representation of $\widehat{\boldsymbol{l}}'$ modulo p. And so when we're given the public key $h(\widehat{\boldsymbol{l}}')$, we can interpret it as a vector (call it $\boldsymbol{y}$), and then try to find a vector $\boldsymbol{b}$ with coefficients at most $10\phi p^{1/m} n \log^2 n$ such that $\boldsymbol{A}\boldsymbol{b} = \boldsymbol{y} (\mathrm{mod} p)$. We will now explain how to use lattice reduction to find such a vector $\boldsymbol{b}$.

We first define a matrix $\boldsymbol{A}' = (\boldsymbol{A}|\boldsymbol{y})$, and then try to find a vector $\boldsymbol{b}'$ such that $\boldsymbol{A}'\boldsymbol{b}' = \mathbf{0} (\mathrm{mod} p)$ where the last coordinate of $\boldsymbol{b}'$ is -1. Notice that this problem is equivalent to the previous one. We now observe that all the vectors $\boldsymbol{b}' \in \mathbb{Z}^{mn+1}$ that satisfy $\boldsymbol{A}'\boldsymbol{b}' = \mathbf{0} (\mathrm{mod} p)$ form an additive group, and thus an integer lattice of dimension $mn+1$. And since we are trying to find a $\boldsymbol{b}'$ with small coordinates, this is akin to finding a short vector in the aforementioned lattice. The basis of this lattice can be constructed in polynomial time (by viewing $\boldsymbol{A}'$ as a linear transformation mapping $\mathbb{Z}^{mn+1}$ to $\mathbb{Z}_p^n$, and computing the kernel of this transformation). And now all we need to do is find a vector in this $mn+1$ dimensional lattice such that all its coordinates are less than $10\phi p^{1/m} n \log^2 n$, and the last coordinate is -1.

Suppose that n is around 512, then $p = n^3 = 2^{27}$, $m = \log n = 9$, and suppose that $\phi = 1$. Thus we need to find a vector whose coordinates are less that $80 n \log^2 n \approx 2^{21}$ in a lattice of dimension $512 * 9 + 1 = 4609$. It's important to notice that this lattice has a vector all of whose coefficients have absolute value at most 1, and all we need is a vector whose coefficients are less than 2^{21}. Such a large vector (relative to the shortest vector) can easily be found by using standard lattice reduction algorithms that find an approximate shortest vector of the lattice. And heuristically, the algorithm can find such a short vector with the added requirement that the last coordinate is -1.

At this point it is unclear exactly how large we would have to set n in order to avoid the above attack, but it is certainly above any parameter that could be useful in practical applications. Nevertheless, we believe that by using the general structure of the scheme presented in this paper as a starting point, it may be possible to construct a practical and secure signature scheme, and this could prove to be a fruitful direction for further research.

Acknowledgements

We would like to thank the anonymous referees for their comments which helped improve the presentation of this paper.

References

1. D. Aharonov and O. Regev. Lattice problems in NP $\cap$ coNP. *Journal of the ACM*, 52(5):749–765, 2005.
2. M. Ajtai. Generating hard instances of lattice problems. *Complexity of Computations and Proofs, Quaderni di Matematica*, 13:1–32, 2004. (Preliminary version in STOC 1996.).
3. M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *STOC*, pages 601–610, 2001.
4. D. Bleichenbacher and U. Maurer. On the efficiency of one-time digital signatures. In *ASIACRYPT*, pages 145–158, 1996.
5. D. Bleichenbacher and U. Maurer. Optimal tree-based one-time digital signature schemes. In *STACS*, pages 363–374, 1996.
6. M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.
7. J. Bos and D. Chaum. Provably unforgeable signatures. In *CRYPTO*, pages 1–14, 1992.
8. R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. *ACM Trans. Inf. Syst. Secur.*, 3(3):161–185, 2000.
9. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
10. I. Dinur. Approximating SVP_∞ to within almost-polynomial factors is NP-hard. *Theor. Comput. Sci.*, 285(1):55–71, 2002.
11. S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. *J. Cryptology*, 9(1):35–67, 1996.
12. R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM Journal on Computing*, 35(1):217–246, 2005.
13. R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In *EUROCRYPT*, pages 123–139, 1999.
14. O. Goldreich and S. Goldwasser. On the limits of nonapproximability of lattice problems. *J. Comput. Syst. Sci.*, 60(3), 2000.
15. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
16. A. Hevia and D. Micciancio. The provable security of graph-based one-time signatures and extensions to algebraic signature schemes. In *Asiacrypt 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 379–396. Springer, 2002.
17. R. Kumar and D. Sivakumar. On polynomial-factor approximations to the shortest lattice vector length. *SIAM J. Discrete Math.*, 16(3):422–425, 2003.

18. V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, pages 144–155, 2006.
19. V. Lyubashevsky, D. Micciancio, C. Peikert, and R.Rosen. Provably secure FFT hashing. Technical report, 2nd NIST Cryptographic Hash Function Workshop, 2006.
20. R. Merkle. A digital signature based on a conventional encryption function. In *CRYPTO*, pages 369–378, 1987.
21. R. Merkle. A certified digital signature. In *CRYPTO*, pages 218–238, 1989.
22. D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 2007. (Special issue on worst-case versus average-case complexity, in print. Available on-line as doi:10.1007/s00037-007-0234-9. Preliminary version in FOCS 2002.).
23. M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.
24. T. Pedersen and B. Pfitzmann. Fail-stop signatures. *SIAM J. Comput.*, 26(2):291–330, 1997.
25. C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, 2006.
26. C. Peikert and A. Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In *STOC*, 2007.
27. J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.
28. M. Szydlo. Merkle tree traversal in log space and time. In *EUROCRYPT*, pages 541–554, 2004.
29. P. van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical Report Technical Report 81-04, University of Amsterdam, http://turing.wins.uva.nl/ peter/, 1981.