Homomorphic Signatures over Binary Fields: Secure Network Coding with Small Coefficients

DAN BONEH*
Stanford University, USA
dabo@cs.stanford.edu

DAVID MANDELL FREEMAN†
Stanford University, USA
dfreeman@cs.stanford.edu

August 20, 2010

Abstract

We propose a new signature scheme that can be used to authenticate data and prevent pollution attacks in networks that use *network coding*. At its core, our system is a homomorphic signature scheme that authenticates vector subspaces of a given ambient space. Our system has several novel properties not found in previous proposals:

- It is the first such scheme that authenticates vectors defined over *binary fields*; previous proposals could only authenticate vectors with large or growing coefficients.
- It is the first such scheme based on the problem of *finding short vectors in integer lattices*, and thus enjoys the worst-case security guarantees common to lattice-based cryptosystems.

Security of our scheme (in the random oracle model) is based on a new hard problem on lattices, called k-SIS, that reduces to standard average-case and worst-case lattice problems.

Our construction gives an example of a cryptographic primitive — homomorphic signatures over $\mathbb{F}_2$ — that can be built using lattice methods, but cannot currently be built using bilinear maps or other traditional algebraic methods based on factoring or discrete-log type problems.

Keywords. Lattice-based cryptography, homomorphic signatures, network coding.

*Supported by NSF.
†Supported by an NSF Mathematical Sciences Postdoctoral Research Fellowship.

1 Introduction

Network coding [3, 16] is a routing method that replaces the traditional "store and forward" paradigm in networks by more intelligent routing that allows intermediate nodes to transform the data in transit. *Linear network coding* [18] refers to a method in which every network node receives a set of data vectors from its peers and sends out a random linear combination of these vectors. The final recipient receives random linear combinations of the originally transmitted vectors and can recover the original data from any set of received vectors that form a full rank matrix. Linear network coding offers robustness and adaptability and has many practical applications in networking [10] and in distributed storage systems such as BitTorrent [13].

Pollution Attacks. If some nodes are malicious and forward vectors not produced using the network coding protocol, then only some of the recipient's vectors are proper linear combinations of the original message vectors. In such a scenario, the recipient has no way of telling which of the received vectors are corrupt and should be ignored during decoding.

Detailed discussion of pollution attacks can be found in [6, 22, 14]. Here we note only that pollution attacks cannot be mitigated by standard signatures or MACs. Clearly, signing the network coded vectors is of no use since recipients do not have the original message vectors and therefore cannot verify the signature. Similarly, signing the entire message prior to transmission does not work. To see why, observe that decoding produces the correct transmitted message only when all vectors being decoded are linear combinations of the original message vectors. A recipient who obtains many vectors where, say, only half are proper linear combinations and the other half are corrupt would need to decode exponentially many subsets until he found a decoded message that is consistent with the signature. We thus see that new integrity mechanisms are needed to mitigate pollution attacks.

Previous Solutions. Several approaches have been proposed to thwart pollution attacks. Of these, some solutions are information theoretic while others are cryptographic. We refer to [6] for a survey of defenses. Here we restrict our attention to cryptographic solutions.

Several authors [8, 17, 22, 6, 11] have devised digital signature schemes for signing a linear subspace V; in the network coding application V is the subspace spanned by the message vectors. Suppose the message vectors have coordinates in a finite field $\mathbb{F}_p$, and let V be the linear space spanned by the k message vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{F}_p^n$ transmitted by the sender. These signature schemes produce a signature σ on V such that σ verifies for every $\mathbf{v} \in V$, but it is difficult to construct a vector $\mathbf{y} \notin V$ such that σ verifies for $\mathbf{y}$. Recipients can thus use the signature σ to reject all received vectors that are not in the subspace V, mitigating the pollution problem.

While the digital signature constructions in [8, 17, 22, 6] are elegant, they require the network coding coefficients to live in a field $\mathbb{F}_p$ where p is the order of a group in which the discrete logarithm problem is infeasible (e.g. $p \approx 2^{160}$). Transmitting each coefficient thus requires 20 bytes, and hence these coefficients add $20k$ bytes to every packet. In addition, multiplications are over the field $\mathbb{F}_p$ and thus computing linear combinations of packets is slow. In linear network coding (without integrity) one can use $p = 2$, in which case the coefficients add only k bits to every packet and linear algebra over $\mathbb{F}_p$ is very fast. Thus a new construction is needed if one wishes to maintain integrity while minimizing the overhead from transmitting coefficients and optimizing processing speed of packets.

The current construction that comes closest to achieving this goal is that of Gennaro et al. [11], which gives a signature scheme for signing a linear subspace where the coefficients are initially binary, but grow over time as more combinations take place in the network.

Our contribution. We construct the first secure network coding signature scheme that authenticates vectors with coordinates in $\mathbb{F}_2$. Our scheme is *homomorphic*: each data vector carries its own signature, and signatures can be "linearly combined" simultaneously with data vectors (cf. [15, 6]).

Our construction is based on the problem of finding short vectors in integer lattices. Specifically, security is based on a new hard problem on lattices, which we call the k-*Small Integer Solutions* (k-SIS) problem. We show that k-SIS reduces to the standard *Small Integer Solution* (SIS) problem, which is known to be as hard as standard worst-case lattice problems [20]. We hope that by articulating the k-SIS problem and reducing it to standard lattice problems, we have provided a new tool for lattice-based cryptography that can be used in other cryptographic constructions.

Our construction gives a nice example of a cryptographic primitive — homomorphic signatures over $\mathbb{F}_2$ — that can be built using lattice methods, but cannot currently be built using bilinear maps or other traditional algebraic methods based on factoring or discrete-log type problems. Furthermore, since there are no known quantum algorithms for solving hard lattice problems, our construction may remain secure even in the presence of a quantum computer.

Our scheme can be extended to authenticate vectors with coefficients in other small fields, including both prime fields and extension fields such as $\mathbb{F}_{2^d}$; a natural field choice is $\mathbb{F}_{256}$ [1].

Overview of the construction. Our construction builds on the signature scheme of Gentry et al. [12], in which signatures are short vectors σ in lattices defined modulo some large integer q. The key idea in our construction is to use short vectors σ in lattices defined modulo $2q$, which allows us to encode different information modulo 2 and modulo q: $\sigma \bmod 2$ encodes information about the vector being signed, while $\sigma \bmod q$ encodes a solution to a hard problem, ensuring that an adversary cannot forge the signature.

The fact that σ is a short *integer* vector ensures that the two parts cannot be attacked independently. Specifically, applying the Chinese remainder theorem to two vectors σ_2 and σ_q that are correct mod 2 and mod q, respectively, does not produce a short integer vector. This property appears to be unique to lattice-based cryptography: if we attempted a similar construction in discrete log groups of order $2q$, we would easily be able to attack the order 2 and order q parts independently.

Concretely, our construction works as follows. Let q be an odd prime. To sign a vector subspace $V = \text{span}(\mathbf{v}_1, \ldots, \mathbf{v}_k)$ of $\mathbb{F}_2^n$, we define a matrix $\mathbf{A}_V \in \mathbb{Z}_{2q}^{m \times n}$ and then sign each basis vector $\mathbf{v}_i$. The signature on $\mathbf{v}_i \in \mathbb{F}_2^n$ is a low-norm vector $\sigma_i \in \mathbb{Z}^m$ such that

$$\mathbf{A}_V \cdot \sigma_i = q \cdot \mathbf{v}_i \mod 2q$$

A signature $\sigma \in \mathbb{Z}^m$ on a vector $\mathbf{y} \in \mathbb{F}_2^n$ is valid if σ has small norm and $\mathbf{A}_V \cdot \sigma = q \cdot \mathbf{y} \bmod 2q$.

Producing such a signature requires knowing a short basis for the matrix $\mathbf{A}_V$; to obtain such a basis we combine the trapdoor generation algorithm of Alwen and Peikert [5] with the basis delegation mechanism of Cash et al. [7].

The homomorphic property of our scheme is now immediate: if we are given arbitrary vector-signature pairs $(\mathbf{u}_j, \sigma_j) \in \mathbb{F}_2^n \times \mathbb{Z}^m$ for $j = 1, \ldots, \ell$, we can create a signature on $\mathbf{u} = \mathbf{u}_1 + \cdots + \mathbf{u}_\ell \in \mathbb{F}_2^n$ by computing $\sigma = \sigma_1 + \cdots + \sigma_\ell \in \mathbb{Z}^m$. Since the σ_j are all valid signatures on the $\mathbf{u}_j$, we see that $\mathbf{A}_V \cdot \sigma = q \cdot \mathbf{u} \bmod 2q$ and σ has low norm (if ℓ is sufficiently small), so σ is a valid signature on $\mathbf{u}$.

To prove security in the model of [6], we need to show that given signatures on basis vectors of V, it is impossible to generate a signature on a vector outside of V. To do so we define the k-SIS problem, which, roughly speaking, is as follows:

> Given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and k short vectors $\mathbf{e}_1, \ldots, \mathbf{e}_k \in \mathbb{Z}^m$ satisfying $\mathbf{A} \cdot \mathbf{e}_i = 0 \bmod q$, find a

short vector $\mathbf{e} \in \mathbb{Z}^m$ satisfying $\mathbf{A} \cdot \mathbf{e} = 0 \bmod q$, such that $\mathbf{e}$ is not in $\mathbb{Q}$-span$(\{\mathbf{e}_1, \ldots, \mathbf{e}_k\})$.

When $k = 0$ this problem is the standard SIS problem.

In Section 5 we show that an adversary that breaks the network coding scheme (defined mod $2q$) in the random oracle model can be used to solve the k-SIS problem (defined mod q). In Section 4 we show that the k-SIS problem is as hard as the SIS problem. Our reduction degrades exponentially in k, which forces us to use a constant-size k if we want our network coding scheme to be provably secure based on worst-case lattice problems. It is a beautiful open problem to give either a tighter reduction to SIS or a direct reduction to worst-case lattice problems.

While our signature scheme provides integrity for network coding using coefficients in $\mathbb{F}_2$ that always remain in $\mathbb{F}_2$, the signatures grow in norm (though not in dimension) as they are combined through hops in the network. As a result, as more combinations are needed, the modulus q must grow. Fortunately, the dependence is well behaved: to support L combinations one need only increase the length of q by roughly $\log L$ bits (i.e., increase q by a factor of L).

Of greater concern is the fact that a signature on a vector $\mathbf{v} \in \mathbb{F}_2^n$ is a vector $\sigma \in \mathbb{Z}^m$ with $m > n \lg q$; thus signatures in our system are larger than the data packets they are verifying. While this renders the scheme impractical for real-world use, it is nonetheless a surprise that homomorphic signatures over $\mathbb{F}_2$ are possible at all.

Outline. Section 2 provides a description of network coding and gives a formal definition and security model (adapted from [6]) for network coding signatures. In Section 3 we review facts about lattices that we will use in our construction and security proof. Section 4 describes the k-SIS problem and gives our reduction of k-SIS to SIS. We present our scheme and prove its security in Section 5. Finally, in Section 6 we consider parameter selection, describe extensions of our scheme to vector spaces over more general fields, and pose some open problems.

Notation. For any integer $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers modulo q. When q is prime, $\mathbb{Z}_q$ is a field and is sometimes denoted $\mathbb{F}_q$. We let $\mathbb{Z}_q^{n \times m}$ denote the set of $n \times m$ matrices with entries in $\mathbb{Z}_q$. We denote matrices by capital boldface letters and vectors by lowercase boldface letters. We say a function $f(n)$ is *negligible* if it is $O(n^{-c})$ for all $c > 0$, and we use $\mathrm{negl}(n)$ to denote a negligible function of n. The function $\lg x$ is the base 2 logarithm of x.

2 Network Coding

To transmit a message using linear network coding [18] the sender first breaks the message into a sequence of k vectors $\hat{\mathbf{v}}_1, \ldots, \hat{\mathbf{v}}_k$ in an n-dimensional linear space $\mathbb{F}_p^n$, where n, k and p are fixed ahead of time. Using $p = 2$ is sufficient so that the entire transmitted message is $n \times k$ bits. The sender transmits these message vectors to its neighboring nodes in the network. As the vectors traverse the network, moving from one node to the next on their way to the destination, the nodes randomly combine the vectors with each other. More precisely, each node in the network creates a random $\mathbb{F}_p$-linear combination of the vectors it receives and transmits the resulting vector to its adjacent nodes. Intended recipients thus receive random linear combinations of the original message vectors. Recipients can recover the original message from any set of k random linear combinations that form a full rank matrix.

For this approach to work, every vector $\hat{\mathbf{y}}$ in the network must carry with it the coefficients $\alpha_1, \ldots, \alpha_k \in \mathbb{F}_p$ that produce $\hat{\mathbf{y}}$ as a linear combination of the original message vectors. To do so, prior to transmission,

the source node augments every message vector $\hat{\mathbf{v}}_i$ with k additional components. The resulting vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$, called *augmented vectors*, are given by:

$$\mathbf{v}_i = (—\hat{\mathbf{v}}_i—, \overbrace{\underbrace{0, \ldots, 0, 1}_{i}, 0, \ldots, 0}^{m}) \quad \in \mathbb{F}_p^{n+k} \tag{2.1}$$

i.e., each original vector $\hat{\mathbf{v}}_i$ is appended with the vector of length k containing a single '1' in the ith position. (A subspace basis $\mathbf{v}_1, \ldots, \mathbf{v}_k$ created in this manner is said to be *properly augmented*.) These augmented vectors are then sent by the source as packets in the network. Observe that if $\mathbf{y} \in \mathbb{F}_p^{n+k}$ is a linear combination of $\mathbf{v}_1, \ldots, \mathbf{v}_m \in \mathbb{F}_p^{n+k}$ then the linear combination coefficients are contained in the last k coordinates of $\mathbf{y}$.

To prevent pollution attacks, the source assigns a file identifier id to the k message vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{F}_p^{n+k}$ and then signs its vector to obtain k signatures $\sigma_1, \ldots, \sigma_k$. It then sends $(\mathbf{v}_i, \sigma_i)$ for $i = 1, \ldots, k$ to its peers. Each peer creates a random linear combination of the received data vectors and uses a $\mathsf{Combine}$ algorithm to generate a signature on the combined data vector from the signatures on the given data vectors. Algorithm Verify is used to verify a signature on a given data vector $\mathbf{y}$. More precisely, a network coding signature scheme is defined as follows.

Definition 2.1 ([6])**.** A *homomorphic network coding signature scheme* is a tuple of probabilistic, polynomial-time algorithms $(\mathsf{Setup}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify})$ with the following functionality:

- $\mathsf{Setup}(n, \mathsf{params})$. On input a security parameter n (in unary) and additional public parameters params that include the dimension N of the ambient space and the dimension k of subspaces to be signed, this algorithm outputs a prime p, a public key pk, and a secret key sk.
- $\mathsf{Sign}(\mathsf{sk}, \mathsf{id}, \mathbf{v})$. On input a secret key sk, a file identifier $\mathsf{id} \in \{0,1\}^n$, and a vector $\mathbf{v} \in \mathbb{F}_p^N$, this algorithm outputs a signature σ.
- $\mathsf{Combine}(\mathsf{pk}, \mathsf{id}, \{(\alpha_i, \sigma_i)\}_{i=1}^{\ell})$. On input a public key pk, a file identifier id, and a set of tuples $\{(\alpha_i, \sigma_i)\}_{i=1}^{\ell}$ with $\alpha_i \in \mathbb{F}_p$, this algorithm outputs a signature σ. (If each σ_i is a valid signature on the vector $\mathbf{v}_i$, then σ should be a signature on $\sum_{i=1}^{\ell} \alpha_i \mathbf{v}_i$.)
- $\mathsf{Verify}(\mathsf{pk}, \mathsf{id}, \mathbf{y}, \sigma)$. On input a public key pk, an identifier $\mathsf{id} \in \{0,1\}^n$, a vector $\mathbf{y} \in \mathbb{F}_p^N$, and a signature σ, this algorithm outputs either 0 (reject) or 1 (accept).

We require that for each $(p, \mathsf{pk}, \mathsf{sk})$ output by $\mathsf{Setup}(n, \mathsf{params})$, the following hold:

1. For all id and all $\mathbf{y} \in \mathbb{F}_p^N$, if $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{id}, \mathbf{y})$ then $\mathsf{Verify}(\mathsf{pk}, \mathsf{id}, \mathbf{y}, \sigma) = 1$.
2. For all $\mathsf{id} \in \{0,1\}^n$ and all sets of triples $\{(\alpha_i, \sigma_i, \mathbf{v}_i)\}_{i=1}^{\ell}$, if it holds that $\mathsf{Verify}(\mathsf{pk}, \mathsf{id}, \mathbf{v}_i, \sigma_i) = 1$ for all i, then

$$\mathsf{Verify}\left(\mathsf{pk},\ \mathsf{id},\ \textstyle\sum_i \alpha_i \mathbf{v}_i,\ \mathsf{Combine}\left(\mathsf{pk},\ \mathsf{id},\ \{(\alpha_i, \sigma_i)\}_{i=1}^{\ell}\right)\right) = 1.$$

In our lattice-based network coding signature scheme, we cannot combine arbitrarily many valid signatures and still guarantee successful verification. We capture this property by saying that the scheme is L*-limited* if correctness property (2) holds for all $\ell \leq L$ whenever σ_i are output by the Sign algorithm.

Security. The security model for network coding signatures allows an adversary to make adaptive signature queries on files of his choosing, with the signer randomly choosing the identifier id for each file queried. The winning condition captures the fact that there are two distinct types of forgeries: a vector-signature pair $(\mathbf{y}^*, \sigma^*)$ that verifies for some file *not* queried to the signer (a *type 1 forgery*), or a pair $(\mathbf{y}^*, \sigma^*)$ that verifies for some file that *was* queried to the signer, but for which $\mathbf{y}^*$ is not a linear combination of the vectors queried (a *type 2 forgery*).

Definition 2.2 ([6])**.** A homomorphic network coding signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify})$ is *secure* if the advantage of any probabilistic, polynomial-time adversary $\mathcal{A}$ in the following security game is negligible in the security parameter n:

Setup: The challenger runs $\mathsf{Setup}(n, \mathsf{params})$ to obtain $(p, \mathsf{pk}, \mathsf{sk})$, and gives p and pk to $\mathcal{A}$.

Queries: Proceeding adaptively, $\mathcal{A}$ specifies a sequence of vector subspaces $V_i \subset \mathbb{F}_p^N$, represented as a sequence of properly augmented basis vectors $\mathbf{v}_{i1}, \ldots, \mathbf{v}_{im}$. For each i, the challenger chooses id_i uniformly from $\{0,1\}^n$ and gives to $\mathcal{A}$ the identifier id_i and the j signatures $\sigma_{ij} \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{id}_i, \mathbf{v}_{ij})$ for $j = 1, \ldots, m$.

Output: $\mathcal{A}$ outputs $\mathsf{id}^* \in \{0,1\}^n$, a *non-zero* vector $\mathbf{y}^* \in \mathbb{F}_p^N$, and a signature σ^*.

The adversary *wins* if $\mathsf{Verify}(\mathsf{pk}, \mathsf{id}^*, \mathbf{y}^*, \sigma^*) = 1$, and either (1) $\mathsf{id}^* \neq \mathsf{id}_i$ for all i (a *type 1 forgery*), or (2) $\mathsf{id}^* = \mathsf{id}_i$ for some i but $\mathbf{y}^* \notin V_i$ (a *type 2 forgery*). The *advantage* $\mathrm{NC\text{-}Adv}[\mathcal{A}, \mathcal{S}]$ of $\mathcal{A}$ is defined to be the probability that $\mathcal{A}$ wins the security game.

3 Lattices, Gaussian Sampling, and Hardness Assumptions

An m-dimensional lattice Λ is a full-rank discrete subgroup of $\mathbb{R}^m$. We will be interested in *integer lattices* Λ, i.e., those whose points have coordinates in $\mathbb{Z}^m$. The lattices we consider consist of vectors either generated by or orthogonal to a certain "arity check" matrix modulo some integer q. More precisely, for any integer $q \geq 2$ and any $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we define

$$\begin{aligned} \Lambda_q^{\perp}(\mathbf{A}) &:= \{\mathbf{e} \in \mathbb{Z}^m : \mathbf{A} \cdot \mathbf{e} = \mathbf{0} \bmod q\} \\ \Lambda_q^{\mathbf{u}}(\mathbf{A}) &:= \{\mathbf{e} \in \mathbb{Z}^m : \mathbf{A} \cdot \mathbf{e} = \mathbf{u} \bmod q\} \\ \Lambda_q(\mathbf{A}) &:= \{\mathbf{e} \in \mathbb{Z}^m : \exists\, \mathbf{s} \in \mathbb{Z}_q^n \text{ with } \mathbf{A}^t \cdot \mathbf{s} = \mathbf{e} \bmod q\}. \end{aligned}$$

The lattice $\Lambda_q^{\mathbf{u}}(\mathbf{A})$ is a coset of $\Lambda_q^{\perp}(\mathbf{A})$; namely, $\Lambda_q^{\mathbf{u}}(\mathbf{A}) = \Lambda_q^{\perp}(\mathbf{A}) + \mathbf{t}$ for any $\mathbf{t}$ such that $\mathbf{A} \cdot \mathbf{t} = \mathbf{u} \bmod q$.

Length of a basis. Let $\mathbf{S}$ be an ordered set of linearly independent (column) vectors $\mathbf{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_k\}$ in $\mathbb{R}^m$. We use the following standard notation:

- $\|\mathbf{S}\|$ denotes the length (using the ℓ_2 norm) of the longest vector in $\mathbf{S}$, i.e. $\|\mathbf{S}\| := \max_i \|\mathbf{s}_i\|$ for $1 \leq i \leq k$.
- $\widetilde{\mathbf{S}} := \{\tilde{\mathbf{s}}_1, \ldots, \tilde{\mathbf{s}}_k\} \subset \mathbb{R}^m$ denotes the Gram-Schmidt orthogonalization of the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_k$ taken in that order. We refer to $\|\widetilde{\mathbf{S}}\|$ as the *Gram-Schmidt norm* of $\mathbf{S}$.

Generating a short basis. The public key for our signature scheme will be a random matrix $\mathbf{A}$, and the secret key will be a basis $\mathbf{S}$ for $\Lambda^{\perp}(\mathbf{A})$ with low Gram-Schmidt norm. We can generate $\mathbf{A}$ and $\mathbf{S}$ using an algorithm of Alwen and Peikert [5], which improves on an algorithm of Ajtai [4].

Theorem 3.1 ([5, Theorem 3.2], with $\delta = 1/3$)**.** *Let q be an integer*[1] *and $m := \lceil 6n \lg q \rceil$. There is a probabilistic polynomial-time algorithm* $\mathsf{TrapGen}(q, n)$ *that outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{n \times m},\ \mathbf{S} \in \mathbb{Z}^{m \times m})$ such that $\mathbf{A}$ is statistically close to a uniform matrix in $\mathbb{Z}_q^{n \times m}$ and $\mathbf{S}$ is a basis for $\Lambda_q^{\perp}\mathbf{A}$ such that $\|\widetilde{\mathbf{S}}\| \leq 30\sqrt{n \lg q}$ with all but negligible probability in n.*

The definition of "statistically close" and other properties of statistical distance that we will need appear in Appendix A. By Lemma A.2, we may assume without loss of generality that the matrix $\mathbf{A}$ generated by $\mathsf{TrapGen}$ has rank n.

Delegating a basis. In our signature scheme, the lattice used to sign a file will need to be derived from two sources: the public key, which is a matrix $\mathbf{A}$ generated using $\mathsf{TrapGen}$, and the file identifier id, which is random. To combine the two, we hash the file identifier to a second matrix and derive a short basis for $\mathbf{A}\|H(\mathsf{id})$. To derive this new basis we use the basis delegation mechanism of Cash et al.'s identity based encryption scheme [7].

Theorem 3.2 ([7, Lemma 3.2])**.** *Let $\mathbf{S} \in \mathbb{Z}^{m \times m}$ be an arbitrary basis of $\Lambda^{\perp}(\mathbf{A})$ for a rank n matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and let $\mathbf{A}' \in Z_q^{n \times m'}$ be arbitrary. There is a deterministic polynomial-time algorithm* $\mathsf{ExtBasis}(\mathbf{S}, \mathbf{B} := \mathbf{A}\|\mathbf{A}')$ *that outputs a basis $\mathbf{T}$ of $\Lambda^{\perp}(\mathbf{B}) \subset \mathbb{Z}^{(m+m') \times (m+m')}$ such that $\|\widetilde{\mathbf{T}}\| = \|\widetilde{\mathbf{S}}\|$.*

Discrete Gaussians. Let L be a subset of $\mathbb{Z}^m$. For any vector $\mathbf{c} \in \mathbb{R}^m$ and any positive parameter $\sigma \in \mathbb{R}_{>0}$, let:

$\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \exp\left(-\pi\|\mathbf{x} - \mathbf{c}\|^2/\sigma^2\right)$ be a Gaussian function on $\mathbb{R}^m$ with center $\mathbf{c}$ and parameter σ,

$\rho_{\sigma,\mathbf{c}}(L) := \sum_{\mathbf{x} \in L} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$ be the discrete integral of $\rho_{\sigma,\mathbf{c}}$ over L, and

$\mathcal{D}_{L,\sigma,c}$ be the discrete Gaussian distribution over L with center $\mathbf{c}$ and parameter σ. In particular, for all $\mathbf{y} \in L$, we have

$$\mathcal{D}_{L,\sigma,\mathbf{c}}(\mathbf{y}) = \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(L)}.$$

For notational convenience, $\rho_{\sigma,\mathbf{0}}$ and $\mathcal{D}_{L,\sigma,\mathbf{0}}$ are abbreviated as ρ_σ and $\mathcal{D}_{L,\sigma}$, respectively.

Sampling from a discrete Gaussian. Gentry et al. [12] construct algorithms for sampling from discrete Gaussians.

Theorem 3.3.

(a) [12, Theorem 4.1] There is a probabilistic polynomial-time algorithm $\mathsf{SampleGaussian}$ *that, given a basis $\mathbf{T}$ of an n-dimensional lattice Λ, a parameter $\sigma \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$, and a center $\mathbf{c} \in \mathbb{R}^n$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$.*

[1] The result in the published version of [5] is stated and proved for odd q, with a note that this restriction can be lifted. The result in the full version has no restriction on q.

(b) [12, Theorem 5.9] There is a probabilistic polynomial-time algorithm SamplePre *that, given a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$*, a basis* $\mathbf{T}$ *of* $\Lambda_q^\perp(\mathbf{A})$*, a parameter* $\sigma \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$*, and a vector* $\mathbf{u} \in \mathbb{Z}^n$*, outputs a sample from a distribution that is statistically close to* $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\sigma}$.

Recall that if $\Lambda_q^{\mathbf{u}}(\mathbf{A})$ is not empty then $\Lambda_q^{\mathbf{u}}(\mathbf{A}) = \mathbf{t} + \Lambda_q^\perp(\mathbf{A})$ for any $\mathbf{t} \in \Lambda_q^{\mathbf{u}}(\mathbf{A})$. Algorithm $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{u}, \sigma)$ simply calls $\mathsf{SampleGaussian}(\mathbf{T}, \sigma, \mathbf{t})$ and subtracts $\mathbf{t}$ from the result. For Gaussians centered at the origin, we use $\mathsf{SampleGaussian}(\mathbf{T}, \sigma)$ to denote $\mathsf{SampleGaussian}(\mathbf{T}, \sigma, \mathbf{0})$. We use the notation $\mathsf{SampleGaussian}(\mathbb{Z}^m, \sigma)$ to denote sampling from the lattice $\mathbb{Z}^m$ with a basis consisting of the m unit vectors.

The smoothing parameter. For an n-dimensional lattice Λ and positive real $\epsilon > 0$, the *smoothing parameter* $\eta_\epsilon(\Lambda)$ of Λ is defined to be the smallest positive s such that $\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \epsilon$ [20]. The key property of the smoothing parameter is that if $\sigma > \eta_\epsilon(\Lambda)$, then every coset of Λ has roughly equal mass. More precisely, for any such σ, if $\epsilon \in (0, 1)$ and $\mathbf{c} \in \mathbb{R}^n$, then we have [12, Lemma 2.7]

$$\tfrac{1-\epsilon}{1+\epsilon} \cdot \rho_\sigma(\Lambda) \leq \rho_{\sigma,\mathbf{c}}(\Lambda) \leq \rho_\sigma(\Lambda). \tag{3.1}$$

For almost all matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, there is a negligible ϵ such that the smoothing parameter $\eta_\epsilon(\Lambda_q^\perp(\mathbf{A}))$ is less than $\omega(\sqrt{\log m})$:

Lemma 3.4 ([12, Lemma 5.3])**.** *Let* q *be a prime and* n, m *integers such that* $m > 2n \lg q$*. Let* f *be some* $\omega(\sqrt{\log m})$ *function. Then there is a negligible function* $\epsilon(m)$ *such that for all but at most a* q^{-n} *fraction of* $\mathbf{A}$ *in* $\mathbb{Z}_q^{n \times m}$ *we have* $\eta_{\epsilon(m)}(\Lambda_q^\perp(\mathbf{A})) < f(m)$.

The following lemma gives a bound on the length of vectors sampled from a Gaussian. The result follows from [20, Lemma 4.4], using [12, Lemma 3.1] to bound the smoothing parameter.

Lemma 3.5 ([20, Lemma 4.4])**.** *Let* $q \geq 2$ *and let* $\mathbf{A}$ *be a matrix in* $\mathbb{Z}_q^{n \times m}$ *with* $m > n$*. Let* $\mathbf{T}$ *be a basis for* $\Lambda_q^\perp(A)$ *and* $\sigma \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log m})$*. Then for* $\mathbf{c} \in \mathbb{R}^m$ *and* $\mathbf{u} \in \mathbb{Z}_q^n$*, we have*

$$\Pr\left[\|\mathbf{x} - \mathbf{c}\| > \sqrt{m}\,\sigma : x \xleftarrow{\mathrm{R}} \mathcal{D}_{\Lambda,\sigma,\mathbf{c}}\right] \;\leq\; \mathrm{negl}(n)$$

Hardness assumption. The security of our signature scheme is based on the problem of finding short vectors in $\Lambda_q^\perp(\mathbf{A})$ for random $\mathbf{A}$. This is known as the *Small Integer Solution* (SIS) problem, and is defined as follows.

Definition 3.6. An instance of the $\mathsf{SIS}_{q,m,\beta}$ problem is a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. A solution to the problem is a nonzero vector $\mathbf{v} \in \mathbb{Z}^m$ such that $\|\mathbf{v}\| \leq \beta$ and $\mathbf{A} \cdot \mathbf{v} = \mathbf{0} \bmod q$ (i.e., $\mathbf{v} \in \Lambda_q^\perp(\mathbf{A})$).

If $\mathcal{A}$ is an algorithm that takes as input a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we define $\mathsf{SIS}\text{-}\mathrm{Adv}[\mathcal{B}, (q, m, \beta)]$ to be the probability that $\mathcal{A}$ outputs a solution to a uniformly random $\mathsf{SIS}_{q,m,\beta}$ problem instance $\mathbf{A}$.

Micciancio and Regev [20] and Gentry et al. [12] show that the (average case) SIS problem is hard assuming worst-case hardness of certain standard lattice problems, such as the *shortest independent vector problem* SIVP and the *shortest vector problem* GapSVP.

4 New Tools

The security of most lattice-based signature schemes depends on the adversary's inability to find a short vector in $\Lambda_q^\perp(\mathbf{A})$ for some public matrix $\mathbf{A}$. However, for homomorphic signatures this criterion is insufficient. Roughly speaking, an adversary in a homomorphic signature scheme will be given several short vectors $\mathbf{e}_1, \ldots, \mathbf{e}_k \in \Lambda_q^\perp(\mathbf{A})$ and must produce a short vector in $\Lambda_q^\perp(\mathbf{A})$ that is not in the span of the $\mathbf{e}_i$. This is a "one-more" variant of the standard SIS problem, analogous to the "one-more discrete logarithm" problem in group-based cryptography (see e.g., [21]). We now formally define the problem.

Definition 4.1. For any integer $k \geq 0$, an instance of the k-$\mathsf{SIS}_{q,m,\beta,\sigma}$ problem is a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a set of k vectors $\mathbf{e}_1, \ldots, \mathbf{e}_k \in \Lambda_q^\perp(\mathbf{A})$ drawn from the distribution $\mathcal{D}_{\Lambda_q^\perp(\mathbf{A}),\sigma}$. A solution to the problem is a nonzero vector $\mathbf{v} \in \mathbb{Z}^m$ such that

1. $\|\mathbf{v}\| \leq \beta$,
2. $\mathbf{A} \cdot \mathbf{v} = \mathbf{0} \bmod q$ (i.e., $\mathbf{v} \in \Lambda_q^\perp(\mathbf{A})$), and
3. $\mathbf{v} \notin \mathbb{Q}\text{-span}(\{\mathbf{e}_1, \ldots, \mathbf{e}_k\})$.

If $\mathcal{A}$ is an algorithm that takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and vectors $\mathbf{e}_i \in \mathbb{Z}^m$ for $i = 1, \ldots, k$, we define k-SIS-$\mathrm{Adv}[\mathcal{B}, (q, m, \beta, \sigma)]$ to be the probability that $\mathcal{A}$ outputs a solution to a k-$\mathsf{SIS}_{q,m,\beta,\sigma}$ problem instance $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$ drawn at random from the appropriate distribution.

When $k = 0$ the k-SIS problem is identical to the SIS problem. The main result of this section is to show that an adversary $\mathcal{A}$ that solves the k-SIS problem in dimension m can be used to solve the SIS problem in dimension $m - k$. More precisely, we have the following:

Theorem 4.2. *Let q be a prime, and let m, β, σ, and k, be polynomial functions of a security parameter n. Suppose that $m \geq 2n \lg q$, $m/k > n$, $\sigma > \omega(\sqrt{\log m})$, $t > \omega(\sqrt{\log n})$, and $q > \sigma \cdot \omega(\sqrt{\log m})$.*

Let $\beta' = \beta \cdot (k^{3/2} + 1)k!(t\sigma)^k$. Let $\mathcal{A}$ be a polynomial-time adversary for the k-$\mathsf{SIS}_{q,m,\beta,\sigma}$ problem. Then there exists a polynomial-time algorithm $\mathcal{B}$ that solves $\mathsf{SIS}_{q,m-k,\beta'}$, such that

$$\mathsf{SIS}\text{-}\mathrm{Adv}[\mathcal{B}, (q, m-k, \beta')] \geq k\text{-}\mathsf{SIS}\text{-}\mathrm{Adv}[\mathcal{A}, (q, m, \beta, \sigma)] - \epsilon,$$

where ϵ is a negligible function of n.

Since the SIS problem is only assumed to be hard for parameters $\beta \in \mathrm{poly}(n)$, the fact that the above reduction degrades exponentially in k means that k must be chosen to be small enough so that β' is still polynomial in n. In our application the parameter σ is $\omega(\sqrt{n})$, which means that k must be chosen to be $O(1)$. In this case, if we take $t = O(\log \sigma)$ and $\beta' = \beta \cdot O(\sigma^k \log^k \sigma)$, then Theorem 4.2 shows that if the $\mathsf{SIS}_{q,m-k,\beta'}$ problem is hard, then the k-$\mathsf{SIS}_{q,m,\beta,\sigma}$ problem is also hard.

The idea of the proof of Theorem 4.2 is as follows: given an SIS challenge $\mathbf{A}' \in \mathbb{Z}_q^{n \times (m-k)}$, we can choose k random vectors $\mathbf{e}_i$ from a Gaussian distribution over $\mathbb{Z}^m$ and append k columns to $\mathbf{A}'$ to create a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ such that the $\mathbf{e}_i$ are in $\Lambda_q^\perp(\mathbf{A})$. If the adversary $\mathcal{A}$ outputs a short vector $\mathbf{e}^* \in \Lambda_q^\perp(\mathbf{A})$ that is $\mathbb{Q}$-linearly independent of the $\{\mathbf{e}_i\}$, then we can compute a short vector $\mathbf{v} \in \Lambda_q^\perp(\mathbf{A})$ with zeroes in the last k entries. Reading off the first $m - k$ entries of $\mathbf{v}$ gives us a short vector in $\Lambda_q^\perp(\mathbf{A}')$.

To turn this idea into a formal proof, we need to show that the tuple $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$ generated in this manner is indistinguishable from a tuple $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$ selected from the distribution of k-SIS challenge instances. To show this, we define two distributions on $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}^{m \times k}$.

For positive integers $m > n > k$, a prime q, and a real $\sigma > 2$, define $\mathbf{DIST}_0(n, m, k, q, \sigma)$ as:

1. For $i = 1, \ldots, k$, sample independent $\mathbf{e}_i \xleftarrow{\mathrm{R}} \mathcal{D}_{\mathbb{Z}^m, \sigma}$.
2. Choose a random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ subject to the condition $\mathbf{A} \cdot \mathbf{e}_i = 0 \bmod q$ for all i.
3. Output $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$.

Define $\mathbf{DIST}_1(n, m, k, q, \sigma)$ as:

1. Sample a random matrix $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$.
2. For $i = 1, \ldots, k$ sample independent $\mathbf{e}_i \xleftarrow{\mathrm{R}} \mathcal{D}_{\Lambda_q^\perp(\mathbf{A}), \sigma}$.
3. Output $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$.

To prove Theorem 4.2, we will use the fact that for appropriate choices of parameters, the distributions $\mathbf{DIST}_0$ and $\mathbf{DIST}_1$ are statistically close.

Theorem 4.3. *Suppose $m \geq 2n \lg q$, $m > 2k$, and $\sigma > \omega(\sqrt{\log m})$. Then the distributions $\mathbf{DIST}_0(n, m, k, q, \sigma)$ and $\mathbf{DIST}_1(n, m, k, q, \sigma)$ are statistically close.*

We will need several preparatory lemmas.

Lemma 4.4. *Suppose $m \geq 2n \lg q$ and $\sigma > \omega(\sqrt{\log m})$. Then for all but a negligible fraction of $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we have*

$$\rho_\sigma(\mathbb{Z}^m) = q^n \cdot \rho_\sigma(\Lambda_q^\perp(\mathbf{A})) \cdot (1 - \mathrm{negl}(n)).$$

Proof. By Lemma 3.4, there is a negligible $\epsilon(n)$ such that $\sigma > \eta_\epsilon(\Lambda_q^\perp(\mathbf{A}))$ for all but a q^{-n} fraction of $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. By (3.1), for all such $\mathbf{A}$ and all $\mathbf{c} \in \mathbb{R}^n$, we have $\rho_{\sigma, \mathbf{c}}(\Lambda_q^\perp(\mathbf{A})) = \rho_\sigma(\Lambda_q^\perp(\mathbf{A}))(1 - \mathrm{negl}(n))$. If we choose a set of coset representatives $\mathbf{c}$ for $\mathbb{Z}^m / \Lambda_q^\perp(\mathbf{A})$, then we have

$$\rho_\sigma(\mathbb{Z}^m) = \sum_{\mathbf{c} \in \mathbb{Z}^m / \Lambda_q^\perp(\mathbf{A})} \rho_{\sigma, \mathbf{c}}(\Lambda_q^\perp(\mathbf{A})) = [\mathbb{Z}^m : \Lambda_q^\perp(\mathbf{A})] \cdot \rho_\sigma(\Lambda_q^\perp(\mathbf{A}))(1 - \mathrm{negl}(n)).$$

If $\mathrm{rank}(\mathbf{A}) = n$, then $[\mathbb{Z}^m : \Lambda_q^\perp(\mathbf{A})] = q^n$ and the result follows. Since $m > 2n$, the fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with $\mathrm{rank}(\mathbf{A}) < n$ is negligible (cf. Lemma A.2).

Lemma 4.5. *Suppose $m \geq 2n \lg q$, $m > 2k$, $\sigma > \omega(\sqrt{\log m})$, and $q > \sigma \cdot \omega(\sqrt{\log m})$ with q prime. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix satisfying the conclusion of Lemma 4.4. Let $\mathbf{e}_i, \ldots, \mathbf{e}_k$ be vectors sampled from $\mathcal{D}_{\Lambda_q^\perp(\mathbf{A}), \sigma}$. Then with overwhelming probability, the set $\{\mathbf{e}_i\}$ has $\mathbb{Z}_q$-rank k.*

Proof. For $i = 1, \ldots, m$ let $\mathbf{E}_i \in \mathbb{Z}_q^{i \times m}$ be the matrix whose rows are the vectors $\mathbf{e}_j \bmod q$ for $j = 1, \ldots, i$. Then the probability that $\mathbf{e}_1, \ldots, \mathbf{e}_k$ are not $\mathbb{Z}_q$-linearly independent is at most

$$\sum_{i=0}^{k-1} \Pr[\mathbf{e}_{i+1} \in \Lambda_q(\mathbf{E}_i)] \leq \frac{1}{\rho_\sigma(\Lambda_q^\perp(\mathbf{A}))} \sum_{i=0}^{k-1} \rho_\sigma(\Lambda_q(\mathbf{E}_i)). \tag{4.1}$$

By [2, Lemma 31], we have $\rho_\sigma(\Lambda_q(\mathbf{E}_i)) \leq \rho_\sigma(\mathbb{Z}^i)/(1 - \epsilon)$, where $\epsilon = 2m \cdot \exp(-(\pi/4)(q/\sigma)^2)$. Using this result and Lemma 4.4, we see that the quantity (4.1) is bounded above by

$$\frac{q^n}{\rho_\sigma(\mathbb{Z}^m)(1 - \epsilon)} \sum_{i=0}^{k-1} \rho_\sigma(\mathbb{Z}^i). \tag{4.2}$$

By [2, Lemma 21] and the assumption $\sigma > \omega(\sqrt{\log m})$, there is a constant $\delta > 0$ such that for all i, we have $\sigma^i \leq \rho_\sigma(\mathbb{Z}^i) \leq \sigma^i(1+\delta)$. Thus the quantity (4.2) is bounded above by

$$\frac{q^n(1+\delta)}{\sigma^m(1-\epsilon)}\left(\frac{\sigma^k-1}{\sigma-1}\right) \leq \delta' \cdot \frac{q^n}{\sigma^{m-k}},$$

for some constant δ'. Since $m \geq 2n \lg q$ and $k < m/2$, this last quantity is less than $\delta' \cdot q^{-n}$ whenever $\sigma > 4$.

Proof of Theorem 4.3. We compute the statistical distance directly. First we note that if the vectors $\{\mathbf{e}_i\}$ chosen in $\mathbf{DIST}_0$ have $\mathbb{Z}_q$-rank ℓ, then there are $q^{n(m-\ell)}$ possible choices for $\mathbf{A}$, with each choice equally likely. We thus see that

$$\Pr\left[X = (\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k) : X \xleftarrow{\mathrm{R}} \mathbf{DIST}_0\right] = \left(\frac{1}{q}\right)^{n(m-\ell)} \prod_{i=1}^{k} \frac{\rho_\sigma(\mathbf{e}_i)}{\rho_\sigma(\mathbb{Z}^m)}.$$

On the other hand, a sample from $\mathbf{DIST}_1$ is nm independent uniform samples from $\mathbb{Z}_q$ and k independent samples from $\mathcal{D}_{\Lambda_q^\perp(\mathbf{A}),\sigma}$, and thus

$$\Pr\left[X = (\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k) : X \xleftarrow{\mathrm{R}} \mathbf{DIST}_1\right] = \left(\frac{1}{q}\right)^{nm} \prod_{i=1}^{k} \frac{\rho_\sigma(\mathbf{e}_i)}{\rho_\sigma(\Lambda_q^\perp(\mathbf{A}))}.$$

Let $S \subset \mathbb{Z}_q^{n\times m}$ be the set of matrices for which the conclusion of Lemma 4.4 holds, and for any $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ let $T_\mathbf{A} \subset \Lambda_q^\perp(\mathbf{A})^k$ be those sets of vectors $\{\mathbf{e}_i\}$ with $\mathbb{Z}_q$-rank k. We first calculate the distance between $\mathbf{DIST}_0$ and $\mathbf{DIST}_1$ when restricted to those tuples for which $\mathbf{A} \in S$ and $\{\mathbf{e}_i\} \in T_\mathbf{A}$:

$$\Delta_0 := \frac{1}{2}\sum_{\mathbf{A}\in S}\sum_{\{\mathbf{e}_i\}\in T_\mathbf{A}} \left| \left(\frac{1}{q}\right)^{n(m-k)} \prod_{i=1}^{k} \frac{\rho_\sigma(\mathbf{e}_i)}{\rho_\sigma(\mathbb{Z}^m)} - \left(\frac{1}{q}\right)^{nm} \prod_{i=1}^{k} \frac{\rho_\sigma(\mathbf{e}_i)}{\rho_\sigma(\Lambda_q^\perp(\mathbf{A}))} \right|.$$

Then by Lemma 4.4, we have

$$\Delta_0 = \frac{1}{2}\sum_{\mathbf{A}\in S} \frac{q^{nk}}{q^{nm}} \cdot \mathrm{negl}(n) \cdot \sum_{\{\mathbf{e}_i\}\in T_\mathbf{A}} \prod_{i=1}^{k} \frac{\rho_\sigma(\mathbf{e}_i)}{\rho_\sigma(\mathbb{Z}^m)}. \tag{4.3}$$

If we relax the restriction on the rank of the $\{\mathbf{e}_i\}$, then the total sum does not decrease, and the numerator of the last sum now includes all terms of the form $\prod_{i=1}^k \rho_\sigma(\mathbf{e}_i)$ with $\{\mathbf{e}_i\} \in \Lambda_q^\perp(\mathbf{A})^k$. Since the expression $(\sum_{\mathbf{e}\in\Lambda_q^\perp(\mathbf{A})} \rho_\sigma(\mathbf{e}))^k$ contains all of these terms and more, we have

$$\Delta_0 \leq \sum_{\mathbf{A}\in S} \frac{q^{nk}}{q^{nm}} \cdot \mathrm{negl}(n) \cdot \left(\sum_{\mathbf{e}\in\Lambda_q^\perp(\mathbf{A})} \frac{\rho_\sigma(\mathbf{e})}{\rho_\sigma(\mathbb{Z}^m)}\right)^k = \sum_{\mathbf{A}\in S} \frac{q^{nk}}{q^{nm}} \cdot \mathrm{negl}(n) \cdot \left(\frac{\rho_\sigma(\Lambda_q^\perp(\mathbf{A}))}{\rho_\sigma(\mathbb{Z}^m)}\right)^k$$

By Lemma 4.4 and since $|S| \leq q^{nm}$, we conclude that $\Delta_0 \leq \mathrm{negl}(n)$.

Next, we claim that

$$\Pr[\mathbf{A} \in S \text{ and } \{\mathbf{e}_i\} \in T_\mathbf{A} : (\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k) \leftarrow \mathbf{DIST}_1] \geq 1 - \mathrm{negl}(n). \tag{4.4}$$

Given this claim, the theorem now follows from statement (2) of Lemma A.1, choosing the set A of the Lemma to be tuples with $\mathbf{A} \in S$ and $\{\mathbf{e}_i\} \in T_{\mathbf{A}}$ and using the fact that $\Delta_0 \leq \mathrm{negl}(n)$.

To show (4.4), it suffices to show that both $\Pr[\mathbf{A} \notin S]$ and $\Pr[\mathbf{A} \in S \text{ and } \{\mathbf{e}_i\} \notin T_{\mathbf{A}}]$ are both negligible for tuples chosen from $\mathbf{DIST}_1$. The first quantity is negligible because all matrices $\mathbf{A}$ are equally likely to be chosen, and by Lemma 4.4 a negligible fraction of all matrices are not in S. The second quantity is negligible by Lemma 4.5.

We can now prove our main theorem.

Proof of Theorem 4.2. Let $\mathcal{A}$ be an algorithm that takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and k vectors $\mathbf{e}_1, \ldots, \mathbf{e}_k \in \mathbb{Z}^m$ and outputs a vector $\mathbf{e} \in \mathbb{Z}^m$. We construct an algorithm $\mathcal{B}$ that takes as input a matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times (m-k)}$ and outputs a vector $\mathbf{w} \in \mathbb{Z}_q^{m-k}$.

Algorithm $\mathcal{B}$ begins by sampling k vectors $\mathbf{e}_i$ at random from a Gaussian over $\mathbb{Z}^m$. It then samples a random matrix $\mathbf{A}$ such that $\mathbf{A} \cdot \mathbf{e}_i = \mathbf{0} \bmod q$ for all i. By Theorem 4.3, the tuple $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$ created in this manner is statistically indistinguishable from a k-SIS challenge. Algorithm $\mathcal{B}$ can thus use the k-SIS solver $\mathcal{A}$ to find a short vector $\mathbf{e} \in \Lambda_q^{\perp}(\mathbf{A})$ and do Gaussian elimination over $\mathbb{Z}$ to find a vector in $\Lambda_q^{\perp}(\mathbf{A})$ whose last k entries are zero.

On a technical level, algorithm $\mathcal{B}$ works as follows:

1. Set $\mathbf{e}_i \leftarrow \mathsf{SampleGaussian}(\mathbb{Z}^m, \sigma)$ for $i = 1, \ldots, n$.
2. Let $\mathbf{E}$ be the $m \times k$ matrix whose columns are the vectors $\mathbf{e}_i$. If $\mathbf{E}$ has $\mathbb{Z}_q$-rank less than k, then abort (the simulation has failed). If the simulation does not fail, then without loss of generality[2] assume that the last k rows of $\mathbf{E}$ are linearly independent mod q.
3. Write $\mathbf{E} = \frac{\mathbf{F}}{\mathbf{G}}$, where $\mathbf{F} \in \mathbb{Z}^{(m-k) \times k}$, and $\mathbf{G} \in \mathbb{Z}^{k \times k}$ has determinant prime to q.
4. Set $\mathbf{U} \leftarrow (-\mathbf{B}) \cdot \mathbf{F} \cdot \mathbf{G}^{-1} \in \mathbb{Z}_q^{n \times k}$.
5. Set $\mathbf{A} \leftarrow \mathbf{B} \| \mathbf{U}$.
6. Run $\mathcal{A}$ on inputs $\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k$, and let $\mathbf{e} \in \mathbb{Z}^m$ be the output.
7. Write $\mathbf{e} = \mathbf{f} \| \mathbf{g}$ with $\mathbf{f} \in \mathbb{Z}^{m-k}$ and $\mathbf{g} \in \mathbb{Z}^k$.
8. Set $\mathbf{x} \leftarrow \det(\mathbf{G}) \cdot \mathbf{G}^{-1} \cdot \mathbf{g} \in \mathbb{Z}^k$.
9. Compute $\mathbf{w} \leftarrow \mathbf{F} \cdot \mathbf{x} - \det(\mathbf{G}) \cdot \mathbf{f} \in \mathbb{Z}^{m-k}$, and output $\mathbf{w}$.

We begin by observing that the selection of the $\mathbf{e}_i$ in Step (1) can be viewed as choosing m vectors from $\mathsf{SampleGaussian}(\mathbb{Z}^k, \sigma)$. If we partition these m vectors into $\lfloor m/k \rfloor$ sets of k vectors (plus some extras), then by [2, Theorem 30] the probability that any one of these sets has $\mathbb{Z}_q$-rank less than k is bounded above by some constant $\delta < 1$. Thus the probability that the matrix $\mathbf{E}$ has rank less than k is bounded above by $\delta^{\lfloor m/k \rfloor}$, which is negligible in n since $m/k \geq n$. Thus the probability that $\mathcal{B}$ aborts in Step (2) is negligible.

Since $\mathbf{E}$ has rank k with overwhelming probability, the distribution $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$ produced by the simulator is statistically close to $\mathbf{DIST}_0(n, m, k, q, \sigma)$. By Theorem 4.3, this distribution is statistically close to that of $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k)$ in a k-SIS challenge. Thus even a computationally unbounded adversary cannot tell if it is interacting with a real k-SIS challenge or with our simulation, except with negligible probability.

To conclude the proof, it suffices to show that $\mathbf{w}$ is a solution to the SIS problem for $\mathbf{B}$; namely, that (a) $\mathbf{w}$ is nonzero, (b) $\mathbf{B} \cdot \mathbf{w} = \mathbf{0} \bmod q$, and (c) $\|\mathbf{w}\| \leq \beta'$.

[2] More precisely, we apply a permutation π to the rows of $\mathbf{E}$ to obtain a matrix $\mathbf{E}'$ whose last k rows have $\mathbb{Z}_q$-rank k, and we apply π^{-1} to the columns of the matrix $\mathbf{A}$ produced in Step (5).

To show (a), first observe that $\mathbf{w} = \mathbf{0}$ if and only if $\mathbf{F} \cdot \mathbf{G}^{-1} \cdot \mathbf{g} = \mathbf{f}$ in $\mathbb{Q}^{m-k}$. Let $\mathbf{y} = \mathbf{G}^{-1} \cdot \mathbf{g} \in \mathbb{Q}^k$. If $\mathbf{w} = \mathbf{0}$, then $\mathbf{f} = \mathbf{F} \cdot \mathbf{y}$ and $\mathbf{g} = \mathbf{G} \cdot \mathbf{y}$, and therefore $\mathbf{E} \cdot \mathbf{y} = \mathbf{e}$. Thus $\mathbf{e}$ is a $\mathbb{Q}$-linear combination of the vectors $\mathbf{e}_1, \ldots, \mathbf{e}_k$, contradicting the fact that $\mathbf{e}$ is a solution the k-SIS challenge.

To show (b), observe that since $\mathbf{e} = \mathbf{f} \| \mathbf{g}$ is a solution to the k-SIS challenge, we have $\mathbf{A} \cdot \mathbf{e} = \mathbf{B} \cdot \mathbf{f} + \mathbf{U} \cdot \mathbf{g} = \mathbf{0} \bmod q$. The construction of $\mathbf{U}$ then implies that $\mathbf{B} \cdot \mathbf{f} = \mathbf{B} \cdot \mathbf{F} \cdot \mathbf{G}^{-1} \cdot \mathbf{g} \bmod q$. It follows that

$$\mathbf{B} \cdot \mathbf{w} = \det(\mathbf{G})(\mathbf{B} \cdot \mathbf{F} \cdot \mathbf{G}^{-1} \cdot \mathbf{g} - \mathbf{B} \cdot \mathbf{f}) = 0 \bmod q.$$

Finally, we bound the length of $\mathbf{w}$. By a standard tail inequality [12, Lemma 4.2], the absolute value of each entry of $\mathbf{E}$ is less than $t\sigma$ with overwhelming probability. Furthermore, since $\|\mathbf{e}\| \leq \beta$ we know that each entry of $\mathbf{e}$ has absolute value bounded by β. Since $\mathbf{G} \cdot \mathbf{x} = \det(\mathbf{G}) \cdot \mathbf{g}$, Cramer's rule [9, Ch. 11, Theorem 26] implies that the ith entry of $\mathbf{x}$ is the determinant of the matrix constructed by replacing the ith column of $\mathbf{G}$ with the vector $\mathbf{g}$. There are $k!$ terms in this determinant, each of which consists of a product of $k-1$ entries from $\mathbf{G}$ and one entry from $\mathbf{g}$. Thus with overwhelming probability, each entry of $\mathbf{x}$ is bounded in absolute value by $\beta \cdot k!(t\sigma)^{k-1}$. It follows that each entry of $\mathbf{F} \cdot \mathbf{x}$ is bounded by $\beta \cdot k \cdot k!(t\sigma)^k$, and thus $\|\mathbf{F} \cdot \mathbf{x}\| \leq \beta \cdot k^{3/2} \cdot k!(t\sigma)^k$. Since $|\det(\mathbf{G})| \leq k!(t\sigma)^k$ with overwhelming probability, we have $\|\det(\mathbf{G}) \cdot \mathbf{f}\| \leq \beta \cdot k!(t\sigma)^k$. We conclude that $\|\mathbf{w}\| \leq \beta \cdot (k^{3/2}+1)k!(t\sigma)^k$ with overwhelming probability.

Our network coding scheme will rely on properties of the signature vectors mod 2. We will need the following result, which shows that for appropriate choices of parameters, a sample from $\mathcal{D}_{\Lambda_q^\perp(\mathbf{A}),\sigma}$ looks uniformly random mod 2.

Proposition 4.6. *Let q be odd, let $m > 2n \lg q$, and let $\sigma > \omega(\sqrt{\log m})$. Suppose $(\mathbf{A}, \mathbf{e}_1, \ldots, \mathbf{e}_k) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}^{m \times k}$ is a tuple selected from the distribution of k-$\mathsf{SIS}_{q,m,\beta,\sigma}$ challenge instances (for any β). Then the distribution of $(\mathbf{A}, \mathbf{e}_1 \bmod 2, \ldots, \mathbf{e}_k \bmod 2) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_2^{m \times k}$ is statistically close to the uniform distribution on $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_2^{m \times k}$.*

Proof. For any $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, let $\Lambda'(A) := \Lambda_q^\perp(\mathbf{A}) \cap (2\mathbb{Z})^m$. We first claim that $\Lambda'(\mathbf{A}) = 2\Lambda_q^\perp(\mathbf{A})$. To see this, observe that $\Lambda'(\mathbf{A})$ is contained in both $\Lambda_q^\perp(\mathbf{A})$ and $(2\mathbb{Z})^m$, and $2\Lambda_q^\perp(\mathbf{A})$ is contained in $\Lambda'(\mathbf{A})$. Let $\ell = \operatorname{rank}(\mathbf{A})$; then $\det(\Lambda_q^\perp(\mathbf{A})) = q^\ell$. Since q is odd, it follows that $q^\ell \mid \det(\Lambda'(\mathbf{A}))$, $2^m \mid \det(\Lambda'(\mathbf{A}))$, and $\det(\Lambda'(\mathbf{A})) \mid 2^m q^\ell$. We conclude that $\det(\Lambda'(\mathbf{A})) = 2^m q^\ell$, and the claim follows.

Since $\Lambda'(\mathbf{A}) = 2\Lambda_q^\perp(\mathbf{A})$, we have $\eta_\epsilon(\Lambda'(\mathbf{A})) = 2\eta_\epsilon(\Lambda_q^\perp(\mathbf{A}))$. By Lemma 3.4, there is a negligible $\epsilon(n)$ such that for all but at most a q^{-n} fraction of $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ we have $\sigma > \eta_\epsilon(\Lambda_q^\perp(\mathbf{A}))$. By [12, Corollary 2.8], for all such $\mathbf{A}$ the distribution of the $\mathbf{e}_i$ is statistically close to uniform over $\Lambda_q^\perp(\mathbf{A})/\Lambda'(\mathbf{A}) = \Lambda_q^\perp(\mathbf{A})/2\Lambda_q^\perp(\mathbf{A}) \cong \mathbb{Z}_2^m$, and the proposition follows for all $\mathbf{A}$ satisfying the bound $\eta_\epsilon(\Lambda_q^\perp(\mathbf{A})) < \sigma$. Since all matrices $\mathbf{A}$ are equally likely and all but a q^{-n} fraction satisfy this bound, the proposition follows.

5 A Network Coding Signature Scheme over $\mathbb{F}_2$

We now describe our network coding signature scheme. Our construction is inspired by the signature scheme of Gentry et al. [12]. In this scheme, signatures are short vectors in $\Lambda_q^{\mathbf{u}}(\mathbf{A})$, where $\mathbf{u}$ is the hash of the message to be signed. The key idea in our construction of homomorphic signatures is to work simultaneously modulo 2 and modulo q. Specifically, a signature on a vector $\mathbf{v} \in \mathbb{F}_2^n$ is a short vector $\mathbf{e} \in \mathbb{Z}^m$ such that $\mathbf{e}$ is in both $\Lambda_q^\perp(\mathbf{A})$ and $\Lambda_2^{\mathbf{v}}(\mathbf{A})$. The mod 2 part ties the signature to the message, while the mod q part ensures that the signature cannot be forged. By the Chinese remainder theorem, such a vector $\mathbf{e}$ is in the lattice $\Lambda_{2q}^{q \cdot \mathbf{v}}$.

In order to be able to sign multiple files, the matrix $\mathbf{A}$ must be different for every message, yet still have a trapdoor that allows us to generate signatures using the SamplePre algorithm. To achieve this, we divide $\mathbf{A}$ into two parts. The left half is a public matrix generated by the TrapGen algorithm, while the right half depends on the identifier of the file being signed. Given the secret basis output by TrapGen, we can use the ExtBasis algorithm to compute a short basis for $\Lambda_{2q}^{\perp}(\mathbf{A})$.

$\mathsf{Setup}(n, \mathsf{params})$. Given a security parameter n and parameters $\mathsf{params} = (N, k, L, m, q, \sigma)$, where $N = n$ is the dimension of vectors to be signed, $k < n$ is the dimension of subspaces to be signed, $L \geq 1$ is a parameter relating to the complexity of the network, $m(n, L) > n$ is an integer, $q(n, L)$ is an odd prime, and $\sigma(n, L)$ is a real number, do the following:

1. Run $\mathsf{TrapGen}(n, m, 2q)$ to generate a matrix $\mathbf{A} \in \mathbb{Z}_{2q}^{n \times m}$ and a basis $\mathbf{T}$ of $\Lambda_{2q}^{\perp}(\mathbf{A})$ such that $\|\widetilde{\mathbf{T}}\| \leq 30\sqrt{n \lg 2q}$.
2. Let $H : \{0,1\}^* \to \mathbb{Z}_{2q}^{n \times m}$ be a hash function, viewed as a random oracle.
3. Output the prime $p = 2$, the public key $\mathsf{pk} \leftarrow (\mathbf{A}, H)$, and the private key $\mathsf{sk} \leftarrow (\mathbf{A}, H, \mathbf{T})$.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{id}, \mathbf{v})$. Given secret key $\mathsf{sk} = (\mathbf{A}, H, \mathbf{T})$, identifier $\mathsf{id} \in \{0,1\}^n$, and a vector $\mathbf{v} \in \mathbb{F}_2^n$, do:

1. Set $\mathbf{B} \leftarrow \mathbf{A} \| H(\mathsf{id}) \in \mathbb{Z}_{2q}^{n \times 2m}$.
2. Set $\mathbf{S} \leftarrow \mathsf{ExtBasis}(\mathbf{T}, \mathbf{B})$ to be a basis for $\Lambda_{2q}^{\perp}(\mathbf{B})$ with $\|\widetilde{\mathbf{S}}\| = \|\widetilde{\mathbf{T}}\|$.
3. Output $\mathbf{e} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{S}, \sigma, q \cdot \mathbf{v})$.

$\mathsf{Combine}(\mathsf{pk}, \mathsf{id}, \{(\alpha_i, \mathbf{e}_i)\}_{i=1}^{\ell})$. Given a public key pk, an identifier id, and $\{(\alpha_i, \mathbf{e}_i)\}_{i=1}^{\ell}$ with $\alpha_i \in \mathbb{F}_2 = \{0,1\}$, output $\mathbf{e} \leftarrow \sum_{i=1}^{\ell} \alpha_i \mathbf{e}_i \in \mathbb{Z}^{2m}$.

$\mathsf{Verify}(\mathsf{pk}, \mathsf{id}, \mathbf{y}, \mathbf{e})$. Given a public key $\mathsf{pk} = (\mathbf{A}, H)$, an identifier id, a signature $\mathbf{e} \in \mathbb{Z}^{2m}$, and a vector $\mathbf{y} \in \mathbb{F}_2^n$, do the following:

1. Set $\mathbf{B} \leftarrow \mathbf{A} \| H(\mathsf{id}) \in \mathbb{Z}_{2q}^{n \times 2m}$.
2. If (a) $\|\mathbf{e}\| \leq L \cdot \sigma\sqrt{m}$ and (b) $\mathbf{B} \cdot \mathbf{e} = q \cdot \mathbf{y} \bmod 2q$, output 1. Otherwise output 0.

Proposition 5.1. *Suppose $\sigma \geq 30\sqrt{n \lg 2q} \cdot \omega(\sqrt{\log n})$. Then the scheme described above is an L-limited homomorphic network coding signature scheme.*

Proof. We must show that the correctness conditions of Definition 2.1 hold, with (2) holding for all $\ell \leq L$. By Theorem 3.3 (b), the vector $\mathbf{e}$ output by the Sign algorithm satisfies $\mathbf{B} \cdot \mathbf{e} = q \cdot \mathbf{v} \bmod 2q$ and is drawn from a distribution statistically close to $\mathcal{D}_{\Lambda_{2q}^{q\mathbf{v}}(\mathbf{B}),\sigma}$. By Lemma 3.5, we have $\|\mathbf{e}\| \leq \sigma\sqrt{m}$ with overwhelming probability. It follows that if $\mathbf{e}$ is output by $\mathsf{Sign}(\mathsf{sk}, \mathsf{id}, \mathbf{v})$, then $\mathsf{Verify}(\mathsf{pk}, \mathsf{id}, \mathbf{v}, \mathbf{e}) = 1$.

Since the network coding coefficients α_i are in $\{0,1\}$, the length of the vector $\mathbf{e}$ output by $\mathsf{Combine}(\mathsf{pk}, \mathsf{id}, \{(\alpha_i, \mathbf{e}_i)\}_{i=1}^{\ell})$ when given signatures $\mathbf{e}_i$ output by Sign is at most $\ell\sigma\sqrt{m}$, so this vector passes verification test (a) whenever $\ell \leq L$. As for verification test (b), suppose we have vectors $\mathbf{v}_i$ such that $\mathsf{Verify}(\mathsf{pk}, \mathsf{id}, \mathbf{v}_i, \mathbf{e}_i) = 1$ for all i. Since q is odd, this implies that $\mathbf{B} \cdot \mathbf{e}_i = 0 \bmod q$ and $\mathbf{v}_i \bmod 2$ for all i. It follows that $\mathbf{B} \cdot \mathbf{e} = 0 \bmod q$ and $\sum \alpha_i \mathbf{v}_i \bmod 2$, and therefore $\mathbf{B} \cdot \mathbf{e} = q \cdot \sum \alpha_i \mathbf{v}_i \bmod 2q$.

5.1 Security

We now prove security of our homomorphic network coding signature scheme, in the random oracle model. Given an adversary that breaks the signature scheme over $\mathbb{Z}_{2q}$, we construct an adversary that simulates the signature scheme and solves the k-SIS problem over $\mathbb{Z}_q$. By Theorem 4.2, this adversary can in turn be used to solve the SIS problem over $\mathbb{Z}_q$.

Our simulation begins by guessing which of the adversary's signature and hash queries will correspond to the file identifier id^* associated with the adversary's forgery and outputting a public key $\mathbf{A}$ derived from the k-SIS challenge matrix. For queries *not* associated with id^*, the simulator "swaps the roles" of the public key and hash function as follows: we use TrapGen to program the random oracle with a matrix $H(\mathsf{id})$ for which we know a short basis, and we use ExtBasis to compute a short basis for $\mathbf{A}\|H(\mathsf{id})$. We can then we compute the signatures as in the real system.

For the query id^*, we construct $H(\mathsf{id}^*)$ so that the k-SIS challenge vectors are valid signatures for the vectors queried by the adversary. We construct the mod q part of $H(\mathsf{id}^*)$ using the fact that valid signatures are elements of $\Lambda_q^\perp(\mathbf{A}\|H(\mathsf{id}^*))$, and we construct the mod 2 part of $H(\mathsf{id}^*)$ using the fact that the k-SIS challenge vectors are statistically close to random mod 2.

With this setup, a forged signature is exactly a solution to the k-SIS problem mod q. We now give the details.

Theorem 5.2. *Let $\mathcal{N}$ be the network coding signature scheme described above. Suppose that $m = \lceil 6n \lg 2q \rceil$ and $\sigma = 30\sqrt{n \lg 2q} \log n$. Let $\beta = L \cdot \sigma \cdot \sqrt{m}$. Then $\mathcal{N}$ is secure in the random oracle model assuming that k-$\mathsf{SIS}_{q,2m,\beta}$ is infeasible.*

In particular, let $\mathcal{A}$ be a polynomial-time adversary as in Definition 2.2. Then there exists a polynomial-time algorithm $\mathcal{B}$ that solves k-$\mathsf{SIS}_{q,2m,\beta,\sigma}$, such that

$$k\text{-SIS-Adv}[\mathcal{B}, (q, 2m, \beta, \sigma)] \geq \frac{1}{Q_s + Q_h + 1} \cdot \text{NC-Adv}[\mathcal{A}, \mathcal{N}] - \frac{Q_s(Q_s + Q_h)}{2^n} - 2^{k-2m} - \epsilon,$$

where Q_s and Q_h are the number of signature and hash queries made by $\mathcal{A}$, respectively, and ϵ is a negligible function of the security parameter n.

Proof. Let $\mathcal{A}$ be an adversary as in Definition 2.2 that makes at most Q_s signature queries and at most Q_h hash queries. We construct an algorithm $\mathcal{B}$ that takes as input a random matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times 2m}$ and k vectors $\mathbf{e}_1^*, \ldots, \mathbf{e}_k^* \leftarrow \mathcal{D}_{\Lambda_q^\perp(\mathbf{B}),\sigma}$, and outputs a short vector $\mathbf{e}^* \in \mathbb{Z}^{2m}$ that is $\mathbb{Q}$-linearly independent of the $\mathbf{e}_i^*$. Algorithm $\mathcal{B}$ simulates the hash function H and the Setup and Sign algorithms of $\mathcal{N}$ as follows:

Setup. $\mathcal{B}$ does the following:

1. Choose random $\mathbf{A}_2 \xleftarrow{\mathrm{R}} \mathbb{F}_2^{n \times m}$.
2. Let $\mathbf{A}_q$ be the left m columns of $\mathbf{B}$.
3. Use the Chinese remainder theorem to compute $\mathbf{A} \in \mathbb{Z}_{2q}^{n \times m}$ such that $\mathbf{A} = \mathbf{A}_2 \bmod 2$ and $\mathbf{A} = \mathbf{A}_q \bmod q$.
4. Choose random $J \xleftarrow{\mathrm{R}} \{1, \ldots, Q_s + Q_h + 1\}$.
5. Output the public key $\mathbf{A}$.

Hash query. When $\mathcal{A}$ requests the value of $H(\mathsf{id})$, algorithm $\mathcal{B}$ does the following:

1. If id has already been queried, return $H(\mathsf{id})$.
2. If id is the Jth hash query, do the following:
   (a) Let $\mathbf{E}$ be the $2m \times k$ matrix whose columns are the vectors $\mathbf{e}_i^*$. If the last m rows of $\mathbf{E}$ have $\mathbb{F}_2$-rank less than k, then abort. (The simulation has failed.)
   (b) If id was chosen by the simulator to answer a signature query, do the following:
      i. Let $\mathbf{v}_1, \ldots, \mathbf{v}_k$ be the vectors queried by the adversary, and let $\mathbf{V}$ be the $n \times k$ matrix whose columns are the vectors $\mathbf{v}_i$.
      ii. Choose a random matrix $\mathbf{U}_2 \leftarrow \mathbb{F}_2^{n \times m}$ such that $(\mathbf{A}_2 \| \mathbf{U}_2) \cdot \mathbf{E} = \mathbf{V} \bmod 2$.[3]

      Otherwise, choose a uniformly random matrix $\mathbf{U}_2 \xleftarrow{\mathrm{R}} \mathbb{F}_2^{n \times m}$.
   (c) Let $\mathbf{U}_q$ be the right m columns of $\mathbf{B}$.
   (d) Use the Chinese remainder theorem to compute $\mathbf{U}_{\mathsf{id}} \in \mathbb{Z}_{2q}^{n \times m}$ such that $\mathbf{U}_{\mathsf{id}} = \mathbf{U}_q \bmod q$ and $\mathbf{U}_{\mathsf{id}} = \mathbf{U}_2 \bmod 2$.
   (e) Return $H(\mathsf{id}) \leftarrow \mathbf{U}_{\mathsf{id}}$.
3. Otherwise, do the following:
   (a) Run $\mathsf{TrapGen}(n, m, 2q)$ to generate a matrix $\mathbf{U}_{\mathsf{id}} \in \mathbb{Z}_{2q}^{n \times m}$ and a short basis $\mathbf{T}_{\mathsf{id}}$ of $\Lambda_{2q}^{\perp}(\mathbf{U}_{\mathsf{id}})$.
   (b) Return $H(\mathsf{id}) \leftarrow \mathbf{U}_{\mathsf{id}}$ and store $\mathbf{T}_{\mathsf{id}}$.

Sign. When $\mathcal{A}$ requests a signature on a file V represented by k vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{F}_2^n$, algorithm $\mathcal{B}$ does the following:

1. Choose a random $\mathsf{id} \xleftarrow{\mathrm{R}} \{0,1\}^n$. If id has already been queried to the hash oracle, then abort. (The simulation has failed.)
2. Simulate a query to $H(\mathsf{id})$. If this query is the Jth distinct hash query, let $\{\mathbf{e}_i\}$ be the k-SIS challenge vectors $\{\mathbf{e}_i^*\}$. Otherwise,
   (a) Set $\mathbf{B} \leftarrow \mathbf{A} \| H(\mathsf{id})$.
   (b) Set $\mathbf{S} \leftarrow \mathsf{ExtBasis}(\mathbf{T}_{\mathsf{id}}, \mathbf{B})$ to be a basis for $\Lambda_{2q}^{\perp}(\mathbf{B})$ with $\|\widetilde{\mathbf{S}}\| = \|\widetilde{\mathbf{T}_{\mathsf{id}}}\|$.
   (c) For $i = 1, \ldots, n$, set $\mathbf{e}_i \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{S}, \sigma, q\mathbf{v}_i) \in \mathbb{Z}^{2m}$.
3. Output $\mathbf{e}_1, \ldots, \mathbf{e}_k$.

Output. Eventually $\mathcal{A}$ outputs a signature $\mathbf{e}^*$, an identifier id^*, and a non-zero vector $\mathbf{y}^* \in \mathbb{F}_2^n$. Algorithm $\mathcal{B}$ outputs $\mathbf{e}^*$.

We first analyze the situations where the simulator can abort without outputting an answer and show that the probability that this happens is negligible, assuming Q_s and Q_h are polynomial in n.

- **Step** (2a) **of hashing.** By Proposition 4.6, the entries of $\mathbf{E}$ are statistically close to uniformly random mod 2. Since the probability that a random matrix in $\mathbb{F}_2^{k \times k}$ is not invertible is $1/2$, the probability that the last m rows of $\mathbf{E}$ have $\mathbb{F}_2$-rank less than k is bounded above by $1/2^{m-n}$ (cf. Lemma A.2).

[3] E.g., by choosing $m - k$ columns of $\mathbf{U}_2$ at random and solving for the remaining ones. The fact that the last m rows of $\mathbf{E}$ have $\mathbb{F}_2$-rank k guarantees that for an appropriate choice of columns, a solution exists and is unique.

- **Step** (1) **of signing.** In this case, the simulator aborts if the identifier chosen for the signature query is the same as an identifier previously queried to the hash oracle. This happens with probability at most $Q_s(Q_h + Q_s)/2^n$.

We next observe that if the simulator does not abort, the distribution of the simulator's outputs are (statistically) indistinguishable from the distribution of the outputs in the real signature scheme, under the assumption that H is a random oracle.

- **The public key.** The matrix $\mathbf{A}$ in the simulation is uniformly random in $\mathbb{Z}_{2q}^{n\times m}$. By Theorem 3.1, the matrix $\mathbf{A}$ in the real system is statistically close to uniform.

- **The output of H (on most queries).** If id is not the Jth hash query, then by Theorem 3.1, the matrices $\mathbf{U}_{\mathsf{id}}$ output by all other hash queries are statistically close to uniform. We defer the remaining case.

- **The output of** Sign **(on most queries).** If id is not the Jth hash query, then Theorem 3.3 (b) shows that the $\mathbf{e}_i$ in both the real construction and the simulation come from a distribution that is statistically close to $\mathcal{D}_{\Lambda_{2q}^{\mathbf{v}_i}(\mathbf{B}),\sigma}$. We have shown above that the matrices $\mathbf{B} = \mathbf{A}\|H(\mathsf{id})$ in the real scheme and simulation come from statistically close distributions. It follows that the output distributions of the signatures are statistically close.

- **The output of H and** Sign **when** id **is the Jth hash query.** By Proposition 4.6, the entries of the vectors $\mathbf{e}_i^*$ are statistically close to uniform mod 2. It follows that $\mathbf{U}_2$ is statistically close to uniform mod 2 given the adversary's view, and therefore $H(\mathsf{id})$ is uniformly random in $\mathbb{Z}_{2q}^{n\times m}$.

  As for the signature, observe that the vectors $\mathbf{e}_i^*$ come from the distribution $\mathcal{D}_{\Lambda_q^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}}),\sigma}$ and $\mathbf{A}\|\mathbf{U}_{\mathsf{id}}$ is constructed so that $\mathbf{e}_i \in \Lambda_2^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}})$. Since $\Lambda_{2q}^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}}) = \Lambda_q^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}}) \cap \Lambda_2^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}})$, the vectors $\mathbf{e}_i^*$ come from the distribution $\mathcal{D}_{\Lambda_{2q}^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}}),\sigma}$, which by Theorem 3.3 (b) is statistically close to the distribution of signatures output by the real system.

Now the probability that the simulator "guesses right"; i.e., that that id^* is the Jth query to the hash function, is $1/(Q_s + Q_h + 1)$. If the simulator guesses right, then $\mathbf{e}^*$ is a valid signature for the nonzero vector $\mathbf{y}^*$ belonging to the file with identifier id^*. In particular, this implies that $\mathbf{e}^*$ is nonzero and $\|\mathbf{e}^*\| < \beta$. Furthermore, our construction implies that $(\mathbf{A}\|\mathbf{U}_{\mathsf{id}^*}) \cdot \mathbf{e}^* = q \cdot \mathbf{y}^* \bmod 2q$, and therefore $\mathbf{e}^*$ is a nonzero vector in $\Lambda_q^\perp(\mathbf{A}\|\mathbf{U}_{\mathsf{id}^*}) = \Lambda_q^\perp(\mathbf{B})$.

It remains to show that $\mathbf{e}^*$ is not a $\mathbb{Q}$-linear combination of the vectors $\mathbf{e}_i^*$. Suppose the contrary; then $\mathbf{e}^* = \sum_{i=1}^k c_i\mathbf{e}_i^*$ for some $c_i \in \mathbb{Q}$. We claim that with overwhelming probability, the c_i all have odd denominator when written in lowest terms. To see this, let d be the least common denominator of the c_i. If some c_i has even denominator, then when we clear denominators we obtain an equation $d\mathbf{e}^* = \sum c_i'\mathbf{e}_i^*$ where d is even and at least one of the $c_i' \in \mathbb{Z}$ is odd. Reducing mod 2, we see that this implies that the $\mathbf{e}_i^*$ are $\mathbb{Z}_2$-linearly dependent. However, by Proposition 4.6 the $\mathbf{e}_i^*$ are statistically close to uniform in $\mathbb{Z}_2^m$, and therefore the probability that the $\mathbf{e}_i^*$ are $\mathbb{Z}_2$-linearly dependent is at most $1/2^{m-k}$, which is negligible (cf. Lemma A.2). This contradicts the hypothesis that some c_i has even denominator with non-negligible probability; in particular, the values $\{c_i \bmod 2\}$ are well-defined with overwhelming probability.

Now suppose that id^* was not produced by the simulator during a signature query (type 1 forgery). Since the vectors $\mathbf{e}_i$ are statistically close to uniform mod 2, their $\mathbb{F}_2$-span is a random subspace of $\mathbb{F}_2^m$ of dimension at most k. Because the adversary has no information about the vectors $\mathbf{e}_i^*$, the probability that $\mathbf{e}^*$ is in the $\mathbb{F}_2$-span of the $\mathbf{e}_i^*$ is at most $1/2^{m-k}$. By our claim above, if $\mathbf{e}^* = \sum c_i\mathbf{e}_i^*$ with $c_i \in \mathbb{Q}$, then this equation holds mod 2 with overwhelming probability, in which case $\mathbf{e}^*$ is in the $\mathbb{F}_2$-span of the $\mathbf{e}_i^*$, a contradiction.

On the other hand, suppose that id^* was produced by the simulator during a signature query (type 2 forgery). Then since $(\mathbf{A}\|\mathbf{U}_{\mathsf{id}^*}) \cdot \mathbf{e}_i^* = \mathbf{v}_i \bmod 2$, we have $\mathbf{y}^* = \sum c_i \mathbf{v}_i \bmod 2$. Thus $\mathbf{y}^*$ is not a valid forgery, and we have again obtained a contradiction.

Corollary 5.3. *Let $\mathcal{N}$ be the network coding signature scheme described above. Suppose that $m = \lceil 6n \lg 2q \rceil$, $\sigma = 30\sqrt{n \lg 2q} \log n$, and $m/k > n$. Let $\beta = L \cdot \sigma \cdot \sqrt{m}$, let $t > \omega(\sqrt{\log n})$, and let $\beta' = \beta \cdot (k^{3/2} + 1)k!(t\sigma)^k$. Then $\mathcal{N}$ is secure in the random oracle model assuming that $\mathsf{SIS}(q, 2m - k, \beta')$ is infeasible.*

In particular, if $k = O(1)$ and $t = \log n$, then we may take $\beta' = O(L \cdot (n \lg 2q)^{k/2+1} (\log n)^{2k+1})$.

Since the SIS problem is only assumed to be hard for $\beta' \in \mathrm{poly}(n)$, our choice of σ forces k to be $O(1)$ to ensure security based on SIS.

Since the distribution of signatures at each layer of the network is different, it may be possible to determine which initial message vectors went into making a given network coded vector. This feature, while noteworthy, does not affect the security of the system.

6 Further Directions

Parameter Selection. We now consider how large the parameter q needs to be in order to guarantee security of our network coding system. We use the result of Gentry, et al. [12, Section 9] which gives a reduction of $\mathsf{SIS}_{q,m,\beta}$ to standard worst-case lattice problems. The reduction requires that m and β be polynomial in n and that

$$q \geq \beta\sqrt{n} \cdot \omega(\sqrt{\log n}). \tag{6.1}$$

By Corollary 5.3, if k is constant and $t = \log n$, then there is some constant c (depending on k) such that our network coding scheme is secure whenever

$$\frac{q}{(\lg 2q)^{k/2+1}} > c(k) \cdot L \cdot n^{(k+3)/2} (\log n)^{2k+2}.$$

Even if the constant c and the network complexity parameter L were equal to 1, for reasonable values of n and k the modulus q must be very large: at least 2^{170} for $n = 1000$ and $k = 10$.

There is, however, a bright spot in this analysis. Since q depends roughly linearly on L, networks with many nodes do not require a much larger q than networks with few nodes. Indeed, setting L to 1000 in the above example increases the size of q by only 10 bits.

The requirement that q be large is due entirely to the fact that the reduction from k-SIS to SIS is exponential in k. If a better reduction could be found and/or k-SIS could be reduced directly to worst-case lattice problems in a way that gave a bound on q similar to (6.1), then $q/\lg q$ would grow like $L \cdot n^{3/2} \log^2 n$, and we could use values of q in the range of 2^{40}.

Extending the System. While we have described a signature scheme that authenticates vectors with coordinates in $\mathbb{F}_2$, the same construction works for any field $\mathbb{F}_p$ where p is a small prime. We simply set $m = \lceil 6n \lg pq \rceil$ and $\sigma = 30\sqrt{n \lg pq} \log n$, and sign a vector $\mathbf{v} \in \mathbb{F}_p^n$ using the lattice $\Lambda_{pq}^{q \cdot \mathbf{v}}(\mathbf{A})$. If p is odd and we identify $\mathbb{F}_p$ with $\{-(p-1)/2, \ldots, (p-1)/2\}$, then the output of $\mathsf{Combine}$ on ℓ vectors can be up to $\ell(p-1)$ times as long as the largest input vector. An argument as in Proposition 5.1 shows that the resulting system is $L/(p-1)$-limited. We must therefore increase L accordingly to allow for as many linear combinations as in the system over $\mathbb{F}_2$.

More interestingly, our system can also be used to verify network coding vectors with coordinates in non-prime fields. Suppose for concreteness that our vectors live in $(\mathbb{F}_{2^d})^n$. If we fix a basis for $\mathbb{F}_{2^d}$ over $\mathbb{F}_2$, then when computing signatures we may view the vectors as elements of $(\mathbb{F}_2)^{nd}$ and compute signatures in exactly the same manner as above. The difference comes when computing linear combinations: multiplying by an element $\alpha \in \mathbb{F}_{2^d}$ corresponds to acting by a matrix $M_\alpha \in \mathbb{F}_2^{d \times d}$. To compute this action on the signature vector $\mathbf{e} \in \mathbb{Z}^m$, we lift M_α to an integer matrix with entries in $\{0, 1\}$ and have it act on each group of d coordinates of $\mathbf{e}$. We see that this action increases the norm of $\mathbf{e}$ by a factor of at most d, so combining ℓ vectors gives an output that is up to ℓd times as long as the largest input vector. By the same argument as above, the system over $\mathbb{F}_{2^d}$ is L/d-limited.

Open Problems. As described above, one important open problem inspired by our construction is to find a tight reduction of k-SIS to worst-case lattice problems, either by improving on the reduction to SIS given by Theorem 4.2 or by a direct argument. An improved reduction would support the use of the k-SIS problem in developing cryptosystems for other applications.

Another problem is to improve the efficiency of our system, which at the moment produces signatures that are too long for real-world applications. One direction to explore is instantiating the construction using *ideal lattices* (see e.g. [19]), which could reduce the size of the public key and possibly that of the signatures as well.

References

[1] S. Agrawal and D. Boneh. "Homomorphic MACs: MAC-based integrity for network coding." In *Proceedings of ACNS '09*, Springer LNCS **5536** (2009), 292–305.

[2] S. Agrawal, D. Boneh, and X. Boyen. "Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE." In *Advances in Cryptology — CRYPTO 2010*, Springer LNCS **6223** (2010), 98–115.

[3] R. Ahlswede, N. Cai, S. Li, and R. Yeung. "Network information flow." *IEEE Transactions on Information Theory* **46** (2000), 1204–1216.

[4] M. Ajtai. "Generating hard instances of the short basis problem." In *ICALP*, ed. J. Wiedermann, P. van Emde Boas, and M. Nielsen, Springer LNCS **1644** (1999), 1–9.

[5] J. Alwen and C. Peikert. "Generating shorter bases for hard random lattices." In *STACS* (2009), 75–86. Full version available at `http://www.cc.gatech.edu/~cpeikert/pubs/shorter.pdf`.

[6] D. Boneh, D. Freeman, J. Katz, and B. Waters. "Signing a linear subspace: Signature schemes for network coding." In *Public-Key Cryptography — PKC '09*, LNCS **5443**. Springer (2009), 68–87.

[7] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. "Bonsai trees, or, how to delegate a lattice basis." In *Advances in Cryptology — EUROCRYPT 2010*, Springer LNCS **6110** (2010), 523–552.

[8] D. Charles, K. Jain, and K. Lauter. "Signatures for network coding." *International Journal of Information and Coding Theory* **1** (2009), 3–14.

[9] D. Dummit and R. Foote. *Abstract Algebra*. 2nd edition. Prentice-Hall, Upper Saddle River, NJ (1999).

[10] C. Fragouli, J.-Y. Le Boudec, and J. Widmer. "Network coding: an instant primer." *SIGCOMM Comput. Commun. Rev.* **36** (2006), 63–68.

[11] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin. "Secure network coding over the integers." In *Public Key Cryptography — PKC '10*, Springer LNCS **6056** (2010), 142–160.

[12] C. Gentry, C. Peikert, and V. Vaikuntanathan. "Trapdoors for hard lattices and new cryptographic constructions." In *STOC*, ed. R. E. Ladner and C. Dwork. ACM (2008), 197–206.

[13] C. Gkantsidis and P. Rodriguez. "Network coding for large scale content distribution." In *IEEE INFOCOM* (2005).

[14] K. Han, T. Ho, R. Koetter, M. Médard, and F. Zhao. "On network coding for security." In *Military Communications Conference (Milcom)* (2007).

[15] R. Johnson, D. Molnar, D. Song, and D. Wagner. "Homomorphic signature schemes." In *Topics in Cryptology — CT-RSA 2002*, Springer LNCS **2271** (2002), 244–262.

[16] R. Koetter and M. Médard. "An algebraic approach to network coding." *IEEE/ACM Transactions on Networking* (2003), 782–795.

[17] M. Krohn, M. Freedman, and D. Mazieres. "On-the-fly verification of rateless erasure codes for efficient content distribution." In *Proc. of IEEE Symposium on Security and Privacy* (2004), 226–240.

[18] S.-Y. R. Li, R. W. Yeung, and N. Cai. "Linear network coding." *IEEE Trans. Info. Theory* **49** (2003), 371–381.

[19] V. Lyubashevsky and D. Micciancio. "Generalized compact knapsacks are collision resistant." In *Proceedings of ICALP '06*, Springer LNCS **4052** (2006), 144–155.

[20] D. Micciancio and O. Regev. "Worst-case to average-case reductions based on Gaussian measures." In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society, Washington, DC, USA (2004), 372–381.

[21] P. Paillier and D. Vergnaud. "Discrete-log-based signatures may not be equivalent to discrete log." In *Advances in Cryptology — ASIACRYPT '05*, Springer LNCS **3788** (2005), 1–20.

[22] F. Zhao, T. Kalker, M. Médard, and K. Han. "Signatures for content distribution with network coding." In *Proc. Intl. Symp. Info. Theory (ISIT)* (2007).

A Probability

Statistical distance. Let X and Y be two random variables taking values in some countable set Ω. Define the *statistical distance*, denoted $\Delta(X;Y)$, to be

$$\Delta(X;Y) := \frac{1}{2}\sum_{s\in\Omega}\left|\Pr[X=s]-\Pr[Y=s]\right|$$

If $X(n)$ and $Y(n)$ are ensembles of random variables, we say X and Y are *statistically close* if $\Delta(X;Y)$ is a negligible function of n.

Lemma A.1. *Let X, Y be random variables taking values in a finite set Ω. Let $A \subseteq \Omega$, let X_A be a random variable taking values in A defined by*

$$\Pr[X_A = s] := \Pr[X = s] / \Pr[X \in A] \quad \textit{for all } s \in A,$$

and let Y_A be defined similarly. Then:

1. *If $\Pr[X \in A] \geq 1 - \epsilon$, then $\Delta(X; X_A) \leq \epsilon$.*
2. *If $\Pr[X \in A] \geq 1 - \epsilon$ and $\Delta(X_A; Y_A) \leq \epsilon$, then $\Delta(X, Y) \leq 4\epsilon$.*

Proof. The first statement is Property (5) of [2, Lemma 12]. For the second statement, we have

$$\begin{aligned}
\Pr[Y \in A] &= \sum_{s \in A} \Pr[Y = s] \\
&= \sum_{s \in A} \Pr[X = s] + \Pr[Y = s] - \Pr[X = s] \\
&\geq \sum_{s \in A} \Pr[X = s] - \Big| \Pr[Y = s] - \Pr[X = s] \Big| \\
&= \Pr[X \in A] - \Delta(X_A; Y_A) \\
&\geq 1 - 2\epsilon.
\end{aligned}$$

It now follows from the first statement that $\Delta(Y; Y_A) \leq 2\epsilon$, and therefore

$$\Delta(X; Y) \leq \Delta(X; X_A) + \Delta(X_A; Y_A) + \Delta(Y_A; Y) \leq 4\epsilon.$$

Random matrices over finite fields.

Lemma A.2. *Let m, n, q be integers with $m > n$ and q prime. Then the probability that a uniformly random matrix $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{F}_q^{n \times m}$ has $\mathbb{F}_q$-rank less than n is at most $1/q^{m-n}$.*

Proof. We view $\mathbf{A}$ as a set of n independent vectors $\mathbf{v}_i \xleftarrow{\mathrm{R}} \mathbb{F}_q^m$. The probability that $\mathbf{A}$ has less than full rank is bounded above by

$$\sum_{i=0}^{n-1} \Pr[\mathbf{v}_{i+1} \in \text{span}(\mathbf{v}_1, \ldots, \mathbf{v}_i)] = \sum_{i=0}^{n-1} \frac{1}{q^{m-i}} < \frac{1}{q^{m-n}}.$$