An extended abstract of this work appears in Advances in Cryptology — EUROCRYPT 2011, *ed. K. Paterson, Springer LNCS* **6632** *(2011), 149–168. This is the full version.*

Homomorphic Signatures for Polynomial Functions

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

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

April 5, 2011

Abstract

We construct the first homomorphic signature scheme that is capable of evaluating multivariate polynomials on signed data. Given the public key and a signed data set, there is an efficient algorithm to produce a signature on the mean, standard deviation, and other statistics of the signed data. Previous systems for computing on signed data could only handle linear operations. For polynomials of constant degree, the length of a derived signature only depends logarithmically on the size of the data set.

Our system uses ideal lattices in a way that is a "signature analogue" of Gentry's fully homomorphic encryption. Security is based on hard problems on ideal lattices similar to those in Gentry's system.

Keywords. Homomorphic signatures, ideals, lattices.

1 Introduction

While recent groundbreaking work has shown how to compute arbitrary functions on *encrypted* data [18, 39, 45], far less is known about computing functions on *signed* data.

Informally, the problem of computing on signed data is as follows. Alice has a numerical data set $m_1, \ldots, m_k$ of size k (e.g., final grades in a course with k students). She independently signs each datum m_i, but before signing she augments m_i with a tag and an index. More precisely, Alice signs the triple ("grades", m_i, i) for $i = 1, \ldots, k$ and obtains k independent signatures $\sigma_1, \ldots, \sigma_k$. Here i is the index of m_i in the data set and the tag "grades" serves as a label that names the data set and binds its members together. For convenience we write $\vec{\sigma} := (\sigma_1, \ldots, \sigma_k)$. The data set and the k signatures are stored on some untrusted remote server.

Later, the server is asked to compute authenticated functions of the data, such as the mean or standard deviation of subsets of the data. To compute a function f, the server uses an algorithm $\mathsf{Evaluate}(\mathsf{pk}, \cdot, f, \vec{\sigma})$ that uses $\vec{\sigma}$ and f to derive a signature σ on the triple

$$\big(\text{ "grades", } m := f(m_1, \ldots, m_k), \langle f \rangle \big), \tag{1.1}$$

where $\langle f \rangle$ is an encoding of the function f, i.e., a string that uniquely describes the function. Note that $\mathsf{Evaluate}$ does not need the original messages — it only acts on signatures. Now the pair (m, σ) can be published and anyone can check that the server correctly applied f to the data set by verifying that σ is a signature on the triple (1.1). The derived signature authenticates both the function f and the result of

*Supported by NSF, DARPA, and AFOSR.
†Supported by an NSF Mathematical Sciences Postdoctoral Research Fellowship.

applying f to the data. The pair (m, σ) can be used further to derive signatures on functions of m and other signed data. We give precise definitions of the system's syntax and security below.

Our focus here is on functions that perform arithmetic operations on the data set, such as mean, standard deviation, and other data mining algorithms. Current methods for computing on signed data can handle only *linear* functions [26, 12, 48, 8, 17, 7]. In these systems, given k independently signed vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$ defined over some finite field $\mathbb{F}_p$, anyone can compute a signature on any vector $\mathbf{v}$ in the $\mathbb{F}_p$-linear span of $\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$, and no one without the secret key can compute a valid signature on a vector $\mathbf{v}$ outside this span. The original motivation for these linear schemes comes from the *network coding* routing mechanism [15].

In this paper we present the first signature system that supports computing *polynomial functions* on signed data. Specifically, our system supports multivariate polynomials of bounded degree. For a constant-degree polynomial with bounded coefficients, the length of a derived signature only depends logarithmically on the size of the data set. Thus, for example, given a signed data set as input, anyone can compute a short signature on the mean, standard deviation, least squares fit, and other functions of arbitrary subsets of the data. Note that computing standard deviation requires only a quadratic multivariate polynomial; other applications, discussed in Section 2.4, may require cubic or higher degree polynomials. While our system intrinsically computes on data defined over a finite field $\mathbb{F}_p$, it can be used to compute on data defined over the integers by choosing a sufficiently large field size p.

Our system's functionality and security derive from properties of certain *integer lattices*. As a "warm-up" to our main result, we describe in Section 4 a linearly homomorphic scheme built from random integer lattices that uses the same underlying ideas as our polynomial scheme. Interestingly, this construction gives a homomorphic system over $\mathbb{F}_2$ that allows linear functions of many more inputs than the best previous such system [7]. In Section 6 we show how replacing the random lattices in the linear scheme with *ideal lattices* leads to a polynomially homomorphic scheme — our main result. In Section 7 we describe an abstract signature scheme that incorporates the properties of both of these instantiations. We prove the security of our abstract scheme and show how our proof specializes to the concrete instantiations.

We note that a trivial solution to computing on signed data is to have the server send the entire data set to the client along with all the signatures and have the client compute the function itself. With our constructions only the output of the function is sent to the client along with a short signature. Beyond saving bandwidth, this approach also limits the amount of information revealed to the client about the data set, as formalized in Section 2.2.

Related work. Before delving into the details of our construction, we mention the related work on non-interactive proofs [29, 44, 22] where the prover's goal is to output a certificate that convinces the verifier that a certain statement is correct. Micali's computationally sound (CS) proofs [29] with Valiant's extension to proofs of knowledge [44] can solve the problem discussed above as follows: Alice signs the pair (τ, D) where D is a data set and τ is a short tag used to name D. She sends D, τ and the signature σ to the server. Later, for some function f, the server publishes $\big(\tau,\ t := f(D),\ \pi\big)$ where π is a short proof of knowledge that there exists a data set D and signature σ such that $t = f(D)$ and σ is a valid signature by Alice on (τ, D). This tuple convinces anyone that t is the result of applying f to the original data set D labeled τ and signed by Alice. Security is proved using Valiant's witness extractor [44] to extract a signature forgery from a cheating server. The construction of π uses the full machinery of the PCP theorem and soundness is in the random oracle model. Gentry and Wichs [21] recently showed that the security of such succinct argument systems cannot be proved in the standard model by a black-box reduction to a falsifiable assumption. We note that computational soundness is sufficient in these settings since the server is given signed data and is therefore already assumed to be computationally bounded.

Our approach replaces the proof π by a simple signature. The server publishes $\big(\tau, \sigma',\ t := f(D)\big)$, where σ' is derived from σ and authenticates both t and f. Constructing σ' is straightforward and takes about the

same amount of work as computing $f(D)$. Moreover, anyone can further compute $t' := g(t) = g(f(D))$ for some function g and use σ' to derive a signature on t' and the function $g(f(\cdot))$. While further computation can also be done with CS proofs [44], it is much simpler with homomorphic signatures.

More recently Goldwasser, Kalai, and Rothblum [22] and Gennaro, Gentry, and Parno [16] as well as [13, 4] show how to outsource computation securely. In [16, 13, 4] and in [22] (in the non-interactive setting) the interaction between the server and the client is tailored to the client and the client uses a secret key to verify the results. In our settings the server constructs a publicly verifiable signature on the result and anyone can verify that signature using Alice's public key.

We also mention another line of related work that studies "redactable" signatures [42, 25, 23, 5, 34, 33, 11, 10, 9, 1, 36]. These schemes have the property that given a signature on a message, anyone can derive signatures on subsets of the message. Our focus here is quite different — we look at computing arithmetic functions on authenticated data, rather than computing on a subset of a single message. We also require that the derived signature explicitly authenticate the computed function f.

Another type of signature that admits computation by a third party is *transitive signatures*, introduced by Rivest and Micali [30, 6, 24, 47, 46]. Transitive signatures are assigned to edges of a graph and enable anyone to derive signatures on paths in the graph.

1.1 Overview of our techniques

The intersection method. Our system uses two n-dimensional integer lattices Λ_1 and Λ_2. The lattice Λ_1 is used to sign the data (e.g., a student's grade or the result of a computation), while the lattice Λ_2 is used to sign a description of the function f applied to the data. The message space for these signatures is $\mathbb{Z}^n/\Lambda_1$, which for the lattices we consider is simply a vector space over the finite field $\mathbb{F}_p$ for some prime p.

A signature in our system is a short vector σ in $\mathbb{Z}^n$ in the intersection of $\Lambda_1 + \mathbf{u}_1$ and $\Lambda_2 + \mathbf{u}_2$ for certain $\mathbf{u}_1, \mathbf{u}_2 \in \mathbb{Z}^n$. In other words, we have $\sigma = \mathbf{u}_1 \bmod \Lambda_1$ and $\sigma = \mathbf{u}_2 \bmod \Lambda_2$. Loosely speaking, this single signature σ "binds" $\mathbf{u}_1$ and $\mathbf{u}_2$ — an attacker cannot generate a new *short* vector σ' from σ such that $\sigma = \sigma' \bmod \Lambda_1$ but $\sigma \neq \sigma' \bmod \Lambda_2$. We refer to this method of jointly signing two vectors $\mathbf{u}_1$ and $\mathbf{u}_2$ as the *intersection method*.

More precisely, let τ be a tag, m be a message, and $\langle f \rangle$ be an encoding of a function f. A signature σ on a triple $(\tau, m, \langle f \rangle)$ is a short vector in $\mathbb{Z}^n$ satisfying $\sigma = m \bmod \Lambda_1$ and $\sigma = \omega_\tau(\langle f \rangle) \bmod \Lambda_2$. Here ω_τ is a hash function defined by the tag τ that maps (encodings of) functions to vectors in $\mathbb{Z}^n/\Lambda_2$. This ω_τ not only must preserve the homomorphic properties of the system, but also must enable simulation against a chosen-message adversary. Note that the Λ_1 component of the signature σ is essentially the same as a Gentry-Peikert-Vaikuntanathan signature [20] on the (unhashed) message m.

It is not difficult to see that these signatures are *additively* homomorphic. That is, let σ_1 be a signature on $(\tau, m_1, \langle f_1 \rangle)$ and let σ_2 be a signature on $(\tau, m_2, \langle f_2 \rangle)$. With an appropriate hash function ω_τ, we can ensure that $\sigma_1 + \sigma_2$ is a signature on $\big(\tau,\ m_1 + m_2,\ \langle f_1 + f_2 \rangle\ \big)$. If we set $\Lambda_1 = (2\mathbb{Z})^n$, we obtain a more efficient linearly homomorphic signature over $\mathbb{F}_2$ than previously known [7].

Now let $g \in \mathbb{Z}[x]$ be a polynomial of degree n and let R be the ring $\mathbb{Z}[x]/(g)$. Then R is isomorphic to $\mathbb{Z}^n$ and ideals in R correspond to integer lattices in $\mathbb{Z}^n$ under the "coefficient embedding." We choose our two lattices Λ_1 and Λ_2 to be prime ideals $\mathfrak{p}$ and $\mathfrak{q}$ in R and a signature on the triple (τ, m, f) to be a short element in R such that $\sigma = m \bmod \mathfrak{p}$ and $\sigma = \omega_\tau(\langle f \rangle) \bmod \mathfrak{q}$. With this setup, let σ_1 and σ_2 be signatures on $(\tau, m_1, \langle f_1 \rangle)$ and $(\tau, m_2, \langle f_2 \rangle)$ respectively. Then for an appropriate hash function ω_τ,

$$\begin{aligned} \sigma_1 + \sigma_2 \quad &\text{is a signature on} \quad \big(\ \tau,\ \ m_1 + m_2,\ \ \langle f_1 + f_2 \rangle\ \big) \quad \text{and} \\ \sigma_1 \cdot \sigma_2 \quad &\text{is a signature on} \quad \big(\ \tau,\ \ m_1 \cdot m_2,\ \ \langle f_1 \cdot f_2 \rangle\ \big)\ . \end{aligned}$$

More generally, we can evaluate any bounded degree polynomial with small coefficients on signatures. In

particular, the quadratic polynomial $v(m_1, \ldots, m_k) := \sum_{i=1}^{k}(km_i - \sum_{i=1}^{k} m_i)^2$ that computes a fixed multiple of the variance can easily be evaluated this way. Anyone can calculate the standard deviation from $v(m_1, \ldots, m_k)$ and k by taking a square root and dividing by k.

Our use of ideal lattices is a signature analogue of Gentry's "somewhat homomorphic" encryption system [18]. Ideal lattices also appear in the lattice-based public key encryption schemes of Stehlé, Steinfeld, Tanaka, and Xagawa [41] and Lyubashevsky, Peikert, and Regev [28] and in the hash functions of Lyubashevsky and Micciancio [27].

Unforgeability. Loosely speaking, a forgery under a chosen message attack is a valid signature σ on a triple $(\tau, m, \langle f \rangle)$ such that $m \neq f(m_1, \ldots, m_k)$, where $m_1, \ldots, m_k$ is the data set signed using tag τ. We show that a successful forger can be used to solve the Small Integer Solution (SIS) problem in the lattice Λ_2, which for random q-ary lattices and suitable parameters is as hard as standard worst-case lattice problems [32]. When Λ_2 is an ideal lattice we can then use the ideal structure to obtain a solution to the Shortest Independent Vectors Problem (SIVP) for the (average case) distribution of lattices produced by our key generation algorithm. As is the case with existing linearly homomorphic signature schemes, our security proofs are set in the random oracle model, where the random oracle is used to simulate signatures for a chosen message attacker.

Privacy. For some applications it is desirable that derived signatures be private. That is, if σ is a signature on a message $m := f(m_1, \ldots, m_k)$ derived from signatures on messages $m_1, \ldots, m_k$, then σ should reveal no information about $m_1, \ldots, m_k$ beyond what is revealed by m and f. Using similar techniques to those in [7], it is not difficult to show that our linearly homomorphic signatures satisfy a privacy property (also defined in [7]) called *weak context hiding*. Demonstrating this property amounts to proving that the distribution obtained by summing independent discrete Gaussians depends only on the coset of the sum.

Interestingly, we can show that our polynomially homomorphic signature is not private. It is an open problem either to design a polynomially homomorphic signature scheme that is also private, or to modify our scheme to make it private.

Length efficiency. We require that derived signatures be not much longer than the original signatures from which they were derived; we define this requirement precisely in Section 2.3. All of our constructions are length efficient.

2 Homomorphic Signatures: Definitions and Applications

Informally, a homomorphic signature scheme consists of the usual algorithms KeyGen, Sign, Verify as well as an additional algorithm Evaluate that "translates" functions on messages to functions on signatures. If $\vec{\sigma}$ is a valid set of signatures on messages $\vec{m}$, then $\mathsf{Evaluate}(f, \vec{\sigma})$ should be a valid signature for $f(\vec{m})$.

To prevent mixing of data from different data sets when evaluating functions, the Sign, Verify, and Evaluate algorithms take an additional short "tag" as input. The tag serves to bind together messages from the same data set. One could avoid the tag by requiring that a new public key be generated for each data set, but simply requiring a new tag for each data set is more convenient.

Formally, a homomorphic signature scheme is as follows:

Definition 2.1. A *homomorphic signature scheme* is a tuple of probabilistic, polynomial-time algorithms (Setup, Sign, Verify, Evaluate) as follows:

- $\mathsf{Setup}(1^n, k)$. Takes a security parameter n and a maximum data set size k. Outputs a public key pk and a secret key sk. The public key pk defines a message space $\mathcal{M}$, a signature space Σ, and a set $\mathcal{F}$ of "admissible" functions $f\colon \mathcal{M}^k \to \mathcal{M}$.

- $\mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. Takes a secret key sk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathcal{M}$ and an index $i \in \{1, \ldots, k\}$, and outputs a signature $\sigma \in \Sigma$.

- $\mathsf{Verify}(\mathsf{pk}, \tau, m, \sigma, f)$. Takes a public key pk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathcal{M}$, a signature $\sigma \in \Sigma$, and a function $f \in \mathcal{F}$, and outputs either 0 (reject) or 1 (accept).

- $\mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma})$. Takes a public key pk, a tag $\tau \in \{0,1\}^n$, a function $f \in \mathcal{F}$, and a tuple of signatures $\vec{\sigma} \in \Sigma^k$, and outputs a signature $\sigma' \in \Sigma$.

Let $\pi_i\colon \mathcal{M}^k \to \mathcal{M}$ be the function $\pi_i(m_1, \ldots, m_k) = m_i$ that projects onto the ith component. We require that $\pi_1, \ldots, \pi_k \in \mathcal{F}$ for all pk output by $\mathsf{Setup}(1^n, k)$.

For correctness, we require that for each $(\mathsf{pk}, \mathsf{sk})$ output by $\mathsf{Setup}(1^n, k)$, we have:

1. For all tags $\tau \in \{0,1\}^n$, all $m \in \mathcal{M}$, and all $i \in \{1, \ldots, k\}$,
   if $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m, i)$, then with overwhelming probability
   $\mathsf{Verify}(\mathsf{pk}, \tau, m, \sigma, \pi_i) = 1$.

2. For all $\tau \in \{0,1\}^n$, all tuples $\vec{m} = (m_1, \ldots, m_k) \in \mathcal{M}^k$, and all functions $f \in \mathcal{F}$, if $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m_i, i)$ for $i = 1, \ldots, k$, then with overwhelming probability

$$\mathsf{Verify}\big(\mathsf{pk},\ \tau,\ f(\vec{m}),\ \mathsf{Evaluate}\big(\mathsf{pk}, \tau, f, (\sigma_1, \ldots, \sigma_k)\big),\ f\big) = 1.$$

We say that a signature scheme as above is *$\mathcal{F}$-homomorphic*, or *homomorphic with respect to $\mathcal{F}$*.

While the Evaluate algorithm in our schemes can take as input derived signatures themselves produced by Evaluate, doing so for a large number of iterations may eventually reach a point where the input signatures to Evaluate are valid, but the output signature is not. Therefore, to simplify the discussion we limit the correctness property to require only that Evaluate produce valid output when given as input signatures $\vec{\sigma}$ produced by the Sign algorithm.

For ease of exposition we describe our systems as if all data sets consist of exactly k items. It is straightforward to apply the systems to data sets of size ℓ for any $\ell \leq k$, simply by interpreting a function on ℓ variables as a function on k variables that ignores the last $k - \ell$ inputs. The definitions of unforgeability and privacy below can be adapted accordingly.

2.1 Unforgeability

The security model for homomorphic signatures allows an adversary to make adaptive signature queries on data sets of his choosing, each containing (up to) k messages, with the signer randomly choosing the tag τ for each data set queried. Eventually the adversary produces a message-signature pair (m^*, σ^*) as well as an admissible function f and a tag τ^*. The winning condition captures the fact that there are two distinct types of forgeries. In a *type 1 forgery*, the pair (m^*, σ^*) verifies for some data set *not* queried to the signer; this corresponds to the usual notion of signature forgery. In a *type 2 forgery*, the pair (m^*, σ^*) verifies for some data set that *was* queried to the signer, but for which m^* does not equal f applied to the messages queried; in other words, the signature authenticates m^* as $f(\vec{m})$ but in fact this is not the case.

Our security model requires that all data in a data set be signed at once; that is, the adversary cannot request signatures on new messages after seeing signatures on other messages in the same data set.

Definition 2.2. A homomorphic signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Evaluate})$ is *unforgeable* if for all k the advantage of any probabilistic, polynomial-time adversary $\mathcal{A}$ in the following game is negligible in the security parameter n:

Setup: The challenger runs $\mathsf{Setup}(1^n, k)$ to obtain $(\mathsf{pk}, \mathsf{sk})$ and gives pk to $\mathcal{A}$. The public key defines a message space $\mathcal{M}$, a signature space Σ, and a set $\mathcal{F}$ of admissible functions $f\colon \mathcal{M}^k \to \mathcal{M}$.

Queries: Proceeding adaptively, $\mathcal{A}$ specifies a sequence of data sets $\vec{m}_i \in \mathcal{M}^k$. For each i, the challenger chooses τ_i uniformly from $\{0,1\}^n$ and gives to $\mathcal{A}$ the tag τ_i and the signatures $\sigma_{ij} \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau_i, m_{ij}, j)$ for $j = 1, \ldots, k$.

Output: $\mathcal{A}$ outputs a tag $\tau^* \in \{0,1\}^n$, a message $m^* \in \mathcal{M}$, a function $f \in \mathcal{F}$, and a signature $\sigma^* \in \Sigma$.

The adversary *wins* if $\mathsf{Verify}(\mathsf{pk}, \tau^*, m^*, \sigma^*, f) = 1$ and either

(1) $\tau^* \neq \tau_i$ for all i (a *type 1 forgery*), or

(2) $\tau^* = \tau_i$ for some i but $m^* \neq f(\vec{m}_i)$ (a *type 2 forgery*).

The *advantage* of $\mathcal{A}$ is the probability that $\mathcal{A}$ wins the security game.

2.2 Privacy

Privacy in our setting means that given signatures on a data set $\vec{m} \in \mathcal{M}^k$, the derived signatures on messages $f_1(\vec{m}), \ldots, f_s(\vec{m})$ do not leak any information about $\vec{m}$ beyond what is revealed by $f_1(\vec{m}), \ldots, f_s(\vec{m})$ for some functions $f_1, \ldots, f_s$ known to the attacker. While we are hiding the original data set $\vec{m}$, we are not hiding the fact the derivation took place. We want the verifier to test that the correct function was applied to the original data.

More precisely, as in [7] we define privacy for homomorphic signatures using a variation of a definition of Brzuska et al. [10]. The definition captures the idea that given signatures on a number of messages derived from two different data sets, the attacker cannot tell which data set the derived signatures came from, and furthermore that this property holds even if the secret key is leaked. We call signatures with this privacy property *weakly context hiding*. The reason for "weak" is that we assume the original signatures on the data set are not public.

The term *context hiding* was first used by Ahn et al. [1] to define a strong notion of privacy that requires derived signatures to be distributed as independent fresh signatures on the same message. This requirement ensures privacy even if the original signatures are exposed. We call the privacy concept defined below *weak context hiding* to clarify that it provides weaker privacy than full context hiding defined in [1].

Definition 2.3. A homomorphic signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Evaluate})$ is *weakly context hiding* if for all k, the advantage of any probabilistic, polynomial-time adversary $\mathcal{A}$ in the following game is negligible in the security parameter n:

Setup: The challenger runs $\mathsf{Setup}(1^n, k)$ to obtain $(\mathsf{pk}, \mathsf{sk})$ and gives pk and sk to $\mathcal{A}$. The public key defines a message space $\mathcal{M}$, a signature space Σ, and a set $\mathcal{F}$ of admissible functions $f\colon \mathcal{M}^k \to \mathcal{M}$.

Challenge: $\mathcal{A}$ outputs $(\vec{m}_0^*, \vec{m}_1^*, f_1, \ldots, f_s)$ with $\vec{m}_0^*, \vec{m}_1^* \in \mathcal{M}^k$. The functions $f_1, \ldots, f_s$ are in $\mathcal{F}$ and satisfy

$$f_i(\vec{m}_0^*) = f_i(\vec{m}_1^*) \qquad \text{for all } i = 1, \ldots, s.$$

In response, the challenger generates a random bit $b \in \{0,1\}$ and a random tag $\tau \in \{0,1\}^n$. It signs the messages in $\vec{m}_b^*$ using the tag τ to obtain a vector $\vec{\sigma}$ of k signatures. Next, for $i = 1, \ldots, s$ the challenger

computes a signature $\sigma_i := \mathsf{Evaluate}(\mathsf{pk}, \tau, f_i, \vec{\sigma})$ on $f_i(\vec{m}_b^*)$. It sends the tag τ and the signatures $\sigma_1, \ldots, \sigma_s$ to $\mathcal{A}$. Note that the functions $f_1, \ldots, f_s$ can be output adaptively after $\vec{m}_0^*, \vec{m}_1^*$ are output.

Output: $\mathcal{A}$ outputs a bit b'.

The adversary $\mathcal{A}$ wins the game if $b = b'$. The *advantage* of $\mathcal{A}$ is the probability that $\mathcal{A}$ wins the game.

Winning the weak context hiding game means that the attacker was able to determine whether the challenge signatures were derived from signatures on $\vec{m}_0^*$ or from signatures on $\vec{m}_1^*$. We say that the signature scheme is *s-weakly context hiding* if the attacker cannot win the privacy game after seeing at most s signatures derived from $\vec{m}_0^*$ or $\vec{m}_1^*$.

2.3 Length efficiency

We say that a homomorphic signature scheme is *length efficient* if for a fixed security parameter n, the length of derived signatures depends only logarithmically on the size k of the data set. More precisely, we have the following:

Definition 2.4. Let $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Evaluate})$ be a homomorphic signature scheme. We say that $\mathcal{S}$ is *length efficient* if there is some function $\mu\colon \mathbb{N} \to \mathbb{R}$ such that for all $(\mathsf{pk}, \mathsf{sk})$ output by $\mathsf{Setup}(1^n, k)$, all $\vec{m} = (m_1, \ldots, m_k) \in \mathcal{M}^k$, all tags $\tau \in \{0,1\}^n$, and all functions $f \in \mathcal{F}$, if

$$\sigma_i \leftarrow \mathsf{Sign}(\mathsf{pk}, \tau, m_i, i) \quad \text{for } i = 1, \ldots, k,$$

then for all $k > 0$, the derived signature $\sigma := \mathsf{Evaluate}\big(\mathsf{pk}, \tau, f, (\sigma_1, \ldots, \sigma_k)\big)$ has bit length

$$\mathrm{len}(\sigma) \leq \mu(n) \cdot \log k$$

with overwhelming probability[1]

A trivial scheme. A trivial construction is one where algorithm $\mathsf{Evaluate}$ outputs the function f along with all of the message-signature pairs in the source data set.[2] The recipient can then evaluate the function f for himself from the originally signed data. Unlike our constructions, this scheme is not length efficient since the length of derived signatures is *linear* in the data set size. Moreover, the scheme is not context hiding since a derived signature reveals everything about the original data set.

2.4 Applications

Before describing our constructions we first examine a few applications of computing on signed data. In the Introduction we discussed applications to computing statistics on signed data, and in particular the examples of mean and standard deviation. Here we discuss more complex data mining algorithms.

Least squares fits. Recall that given an integer $d \geq 0$ and a data set $\{(x_i, y_i)\}_{i=1}^k$ consisting of k pairs of real numbers, the *degree d least squares fit* is a polynomial $f \in \mathbb{R}[x]$ of degree d that minimizes the quantity $\sum_{i=1}^k (y_i - f(x_i))^2$. The vector of coefficients of f is denoted by $\vec{f}$ and is given by the formula

$$\vec{f} = (X^T X)^{-1} X^T \vec{y} \in \mathbb{R}^{d+1},$$

[1] See Section 3 for a technical definition of "overwhelming."

[2] Strictly speaking, $\mathsf{Evaluate}$ is not given the source messages as input. However, the signing algorithm in a trivial construction can embed a message inside the signature on that message so that $\mathsf{Evaluate}$ obtains all the source messages from their signatures.

where $X \in \mathbb{R}^{k\times(d+1)}$ is the *Vandermonde matrix* of the x_i, whose jth column is the vector $(x_1^{j-1}, \ldots, x_k^{j-1})$, and $\vec{y}$ is the column vector $(y_1, \ldots, y_k)$. The degree d is usually small, e.g. $d = 1$ for a least squares fit with a line.

Using homomorphic signatures, a server can be given a set of individually signed data points and derive from it a signature on the least squares fit f (or more precisely, a signature on the vector of coefficients $\vec{f}$). If the signature is length efficient, then for fixed d the length of the derived signature depends only logarithmically on the number of data points k. If the signatures are private, then the derived signature on f reveals nothing about the original data set beyond what is revealed by f.

We consider two types of data sets. In the first type, the x-coordinates are universal constants in $\mathbb{Z}$ and need not be signed. For example, the data set might contain the temperature on each day of the year, in which case the x-coordinates are simply the days of the year and need not be explicitly included in the data. Only the y-coordinates are signed. Then the least squares fit is simply a linear function of $\vec{y}$, namely $\vec{f} := A \cdot \vec{y}$ for some fixed matrix A. The signature on $\vec{f}$ can thus be derived using any *linearly* homomorphic signature scheme. One complication is that our schemes sign integer values, while the matrix A will in general contain non-integer values even if the data are pairs of integers. However, if we factor out the denominator and write $A = A'/c$ for some integer matrix A' and scalar c, then the linearly homomorphic signature enables the server to derive a signature on $A'\vec{y}$, and anyone can divide $A'\vec{y}$ by c to obtain $\vec{f}$.

The second type of data set is one in which both the x-coordinate and the y-coordinate are signed. More precisely, an integer data set $\{(x_i, y_i)\}_{i=1}^k$ is signed by signing all $2k$ values separately (but with the same tag) to obtain $2k$ signatures. The server is given the data set and these $2k$ signatures. Using a homomorphic signature scheme for polynomials of degree $d^2 + d + 1$ over the integers, the server can derive a signature on the least squares fit as follows. First, the server derives signatures on the entries of the integer matrix $B := X^T X \in \mathbb{Z}^{(d+1)\times(d+1)}$. A straightforward computation shows that the degrees of the entries of B in the variables x_i are

$$\deg(B_{ij}) = \begin{pmatrix} 0 & 1 & \cdots & d \\ 1 & 2 & \cdots & d+1 \\ \vdots & \vdots & \ddots & \vdots \\ d & d+1 & \cdots & 2d \end{pmatrix}, \tag{2.1}$$

Now let $c := \det(B)$. It follows from (2.1) that each term in the computation of c has degree $d(d+1)$ in the variables x_i, and thus the server can compute a signature on c if the signature scheme is homomorphic for polynomials of degree $d^2 + d$.

Next, the matrix cB^{-1}, known as the *adjugate* of B [40, p. 117], is an integer matrix whose (i, j)th entry is $(-1)^{i+j}$ times the determinant of the (j, i)th minor of B. The degrees of the entries of cB^{-1} in the variables x_i are thus

$$\deg((cB^{-1})_{ij}) = \begin{pmatrix} d^2+d & d^2+d-1 & \cdots & d^2 \\ d^2+d-1 & d^2+d-2 & \cdots & d^2-1 \\ \vdots & \vdots & \ddots & \vdots \\ d^2 & d^2-1 & \cdots & d^2-d \end{pmatrix}.$$

Since the jth row of X^T has degree $j - 1$ in the x_i, we see that the entries of the matrix $cB^{-1}X^T$ have degree at most $d^2 + d$ in the x_i. Thus if the signature scheme is homomorphic for polynomials of degree $d^2 + d + 1$, the server can compute a signature on each entry of the integer vector $\mathbf{v} := cB^{-1}X^T\vec{y} \in \mathbb{Z}^{d+1}$.

The signature on the least squares fit consists of the signature on c and the signatures on the entries of $\mathbf{v}$. Anyone can then compute $\vec{f} := \mathbf{v}/c$ to obtain the least squares fit $\vec{f}$. The server ends up publishing a vector $\mathbf{v} \in \mathbb{Z}^{d+1}$ and a scalar $c \in \mathbb{Z}$ along with these $d + 2$ signatures. In particular, for a least squares fit using a line it suffices for the signature to be homomorphic for cubic polynomials; the final signature is a signature

on three integer scalars. Note that this method is not private since it reveals c, which is more information about the original data set than is revealed by $\vec{f}$ alone.

In summary, linearly homomorphic signatures are sufficient when the x-coordinates are absolute constants and polynomially homomorphic signatures are needed for general data sets.

More advanced data mining. If we had fully homomorphic signatures (i.e., supporting arbitrary computation on signed data), then an untrusted server could run more complex data mining algorithms on the given data set. For example, given a signed data set, the server could publish a signed decision tree (e.g., as generated by the ID3 algorithm [37]). Length efficiency means that the length of the resulting signature depends only logarithmically on the size of the data set. If the signatures were private, then publishing the signed decision tree would leak no other information about the original data set.

3 Preliminaries

Notation. For any integer $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers modulo q. If q is prime, $\mathbb{Z}_q$ is a field and is denoted by $\mathbb{F}_q$. We let $\mathbb{Z}_q^{\ell \times n}$ denote the set of $\ell \times n$ matrices with entries in $\mathbb{Z}_q$. 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. We say $f(n)$ is *polynomial* if it is $O(n^c)$ for some $c > 0$, and we use $\mathrm{poly}(n)$ to denote a polynomial function of n. We say an event occurs with *overwhelming probability* if its probability is $1 - \mathrm{negl}(n)$. The function $\lg x$ is the base 2 logarithm of x.

Lattices. An *n-dimensional lattice* is a full-rank discrete subgroup of $\mathbb{R}^n$. We will often be interested in *integer lattices*, i.e., those whose points have coordinates in $\mathbb{Z}^n$. Among these lattices are the "q-ary" lattices defined as follows: for any integer $q \geq 2$ and any $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$, we define[3]

$$\begin{aligned} \Lambda_q^{\perp}(\mathbf{A}) &:= \{\mathbf{e} \in \mathbb{Z}^n : \mathbf{A} \cdot \mathbf{e} = \mathbf{0} \bmod q\} \\ \Lambda_q^{\mathbf{u}}(\mathbf{A}) &:= \{\mathbf{e} \in \mathbb{Z}^n : \mathbf{A} \cdot \mathbf{e} = \mathbf{u} \bmod q\}. \end{aligned}$$

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 $\mathbf{A} \cdot \mathbf{t} = \mathbf{u} \bmod q$.

The Gram-Schmidt norm of a basis. Let $\mathbf{S}$ be a set of 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 of the longest vector in $\mathbf{S}$, i.e., $\max_{1 \leq i \leq k} \|\mathbf{s}_i\|$.
- $\widetilde{\mathbf{S}} := \{\tilde{\mathbf{s}}_1, \ldots, \tilde{\mathbf{s}}_k\} \subset \mathbb{R}^m$ denotes the Gram-Schmidt orthogonalization of the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_k$.

We refer to $\|\widetilde{\mathbf{S}}\|$ as the *Gram-Schmidt norm* of $\mathbf{S}$. Micciancio and Goldwasser [31] showed that a full-rank set $\mathbf{S}$ in a lattice Λ can be converted into a basis $\mathbf{T}$ for Λ with an equally low Gram-Schmidt norm:

Lemma 3.1 ([31, Lemma 7.1])**.** *Let Λ be an m-dimensional lattice. There is a deterministic polynomial-time algorithm that, given an arbitrary basis of Λ and a full-rank set $\mathbf{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_m\}$ in Λ, returns a basis $\mathbf{T}$ of Λ satisfying*

$$\|\widetilde{\mathbf{T}}\| \leq \|\widetilde{\mathbf{S}}\| \quad \textit{and} \quad \|\mathbf{T}\| \leq \|\mathbf{S}\|\sqrt{m}/2.$$

[3] We use matrices of dimension $\ell \times n$, rather than the more common $n \times m$, in order to ensure that all of the lattices we discuss are n-dimensional, as well as to reserve the variable m for messages being signed.

Ajtai [2] and later Alwen and Peikert [3] showed how to sample an essentially uniform matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ along with a basis $\mathbf{S}$ of $\Lambda_q^\perp(\mathbf{A})$ with low Gram-Schmidt norm.

Theorem 3.2 ([3, Theorem 3.2] with $\delta = 1/3$)**.** *Let q, ℓ, n be positive integers with $q \geq 2$ and $n \geq 6\ell \lg q$. There is a probabilistic polynomial-time algorithm* $\mathsf{TrapGen}(q, \ell, n)$ *that outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{\ell \times n},\ \mathbf{S} \in \mathbb{Z}^{n \times n})$ such that $\mathbf{A}$ is statistically close to a uniform in $\mathbb{Z}_q^{\ell \times n}$ and $\mathbf{S}$ is a basis for $\Lambda_q^\perp(\mathbf{A})$, satisfying*

$$\|\widetilde{\mathbf{S}}\| \leq O(\sqrt{\ell \log q}) \quad \textit{and} \quad \|\mathbf{S}\| \leq O(\ell \log q)$$

with all but negligible probability in ℓ.

Gaussian distributions. Let L be a subset of $\mathbb{Z}^n$. For any vector $\mathbf{c} \in \mathbb{R}^n$ 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 $\mathbb{R}^n$ with center $\mathbf{c}$ and parameter σ. Let $\rho_{\sigma,\mathbf{c}}(L) := \sum_{\mathbf{x} \in L} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$ be the discrete integral of $\rho_{\sigma,\mathbf{c}}$ over L, and let $\mathcal{D}_{L,\sigma,\mathbf{c}}$ 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.

Gentry, Peikert, and Vaikuntanathan [20] construct algorithms for sampling from discrete Gaussians:

Theorem 3.3.

(a) [20, 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 $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$.*

(b) [20, Theorem 5.9] There is a probabilistic polynomial-time algorithm $\mathsf{SamplePre}$ *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 vector $\mathbf{t} \in \mathbb{R}^n$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda+\mathbf{t},\sigma}$.*

Algorithm $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{t}, \sigma)$ simply calls $\mathsf{SampleGaussian}(\mathbf{T}, \sigma, -\mathbf{t})$ and adds $\mathbf{t}$ to 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 $\mathsf{SampleGaussian}(\mathbb{Z}^n, \sigma)$ to denote sampling from the lattice $\mathbb{Z}^n$ 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 $\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \epsilon$ [32]. The key property of the smoothing parameter is that if $\sigma > \eta_\epsilon(\Lambda)$, then every coset of Λ has roughly equal mass under the distribution $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}$. The following lemma bounds the smoothing parameter:

Lemma 3.4 ([20, Lemma 3.1])**.** *For any n-dimensional lattice Λ and any basis $\mathbf{B}$ of Λ, there is a negligible $\epsilon(n)$ for which $\eta_\epsilon(\Lambda) \leq \|\widetilde{\mathbf{B}}\| \cdot \omega(\sqrt{\log n})$.*

For random q-ary lattices $\Lambda_q^\perp(\mathbf{A})$ (with q prime) there is a much better bound:

Lemma 3.5 ([20, Lemma 5.3])**.** *Let q be a prime and ℓ, n be integers such that $n \geq 2\ell \lg q$. Let f be some $\omega(\sqrt{\log n})$ function. Then there is a negligible function $\epsilon(n)$ such that for all but at most a $q^{-\ell}$ fraction of $\mathbf{A}$ in $\mathbb{Z}_q^{\ell \times n}$ we have $\eta_\epsilon(\Lambda_q^\perp(\mathbf{A})) < f(n)$.*

The following lemma gives a bound on the length of vectors sampled from a discrete Gaussian. The result follows from [32, Lemma 4.4], using Lemma 3.4 to bound the smoothing parameter.

Lemma 3.6 ([32, Lemma 4.4])**.** *Let Λ be an n-dimensional lattice, let $\mathbf{T}$ be a basis for Λ, and suppose $\sigma \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$. Then for any $\mathbf{c} \in \mathbb{R}^n$ we have*

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

Sums of discrete Gaussians. The privacy property of our linearly homomorphic scheme will depend on the fact that the distribution of a linear combination of samples from a discrete Gaussian is itself a discrete Gaussian. The following theorem makes this statement precise.

Theorem 3.7 ([7, Theorem 4.13]). *Let $\Lambda \subseteq \mathbb{Z}^m$ be a lattice and $\sigma \in \mathbb{R}$. For $i = 1, \ldots, k$ let $\mathbf{t}_i \in \mathbb{Z}^m$ and let X_i be mutually independent random variables sampled from $\mathcal{D}_{\Lambda+\mathbf{t}_i,\sigma}$. Let $\mathbf{c} = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, and define*

$$g := \gcd(c_1, \ldots, c_k), \qquad \mathbf{t} := \textstyle\sum_{i=1}^{k} c_i \mathbf{t}_i.$$

Suppose that $\sigma > \|\mathbf{c}\| \cdot \eta_\epsilon(\Lambda)$ for some negligible ϵ. Then $Z = \sum_{i=1}^{k} c_i X_i$ is statistically close to $\mathcal{D}_{g\Lambda+\mathbf{t},\|\mathbf{c}\|\sigma}$.

We will also need a generalization of Theorem 3.7 to multivariate distributions. For a matrix $A \in \mathbb{Z}^{s\times k}$ of rank s, define

$$\mu(A) := \sqrt{\det(A\, A^{\mathsf{T}})}\,.$$

If A has rank less than s then to define $\mu(A)$ we first define a matrix $\bar{A}$ as follows: scan the rows of A from first to last adding a row to $\bar{A}$ only if it is linearly independent of the rows already in $\bar{A}$. Then $\bar{A}$ is a maximal subset of linearly independent rows of A, and we define $\mu(A) := \mu(\bar{A})$.

Theorem 3.8 ([7, Theorem 4.14]). *Let Λ be an m-dimensional lattice, $A \in \mathbb{Z}^{s\times k}$ and $T \in \mathbb{Z}^{k\times m}$. Let X be a $k \times m$ random matrix whose rows are mutually independent and whose ith row is distributed as $\mathcal{D}_{\Lambda+\mathbf{t}_i,\sigma}$ where $\mathbf{t}_i$ is the ith row of T. Furthermore, suppose that $\sigma > \mu(A)\ \eta_\epsilon(\Lambda)$ for some negligible ϵ.*

Then the random variable AX is statistically close to a distribution parametrized by Λ, σ, A, and AT.

Theorem 3.8 shows that AX depends on the matrix T only via the quantity AT. Consequently, if X_1 and X_2 are random variables defined as X in the theorem, but using matrices T_1 and T_2 respectively, then AX_1 is statistically close to AX_2 whenever $AT_1 = AT_2$. Theorem 3.7 is a special case where $s = 1$.

Uniformity of discrete Gaussians. Let Λ_1 and Λ_2 be two lattices in $\mathbb{Z}^n$ such that $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$. We will need the following fact, which shows that a discrete Gaussian sampled from Λ_1 is close to uniform in $\mathbb{Z}^n/\Lambda_2$.

Lemma 3.9. *Let Λ_1, Λ_2 be n-dimensional lattices. Then for any $\epsilon \in (0, \frac{1}{2})$, and $\sigma \geq \nu_\epsilon(\Lambda_1 \cap \Lambda_2)$, and any $\mathbf{c} \in \mathbb{R}^n$, the distribution $(\mathcal{D}_{\Lambda_1,\sigma,\mathbf{c}} \bmod \Lambda_2)$ is within statistical distance at most 2ϵ of the uniform distribution over $(\Lambda_1 + \Lambda_2)/\Lambda_2$.*

Proof. By [20, Corollary 2.8], the distribution of $(\mathcal{D}_{\Lambda_1,\sigma,\mathbf{c}} \bmod \Lambda_1 \cap \Lambda_2)$ is within statistical distance at most 2ϵ of uniform over $\Lambda_1/(\Lambda_1 \cap \Lambda_2)$. By the Second Isomorphism Theorem for abelian groups [14, Ch. 3, Theorem 18], $\Lambda_1/(\Lambda_1 \cap \Lambda_2) \cong (\Lambda_1 + \Lambda_2)/\Lambda_2$. Hence, there is a one-to-one correspondence between cosets of $\Lambda_1 \cap \Lambda_2$ in Λ_1 and cosets of Λ_2 inside of $\Lambda_1 + \Lambda_2$.

Now, for some $\mathbf{v} \in \Lambda_1$ let $I_\mathbf{v} \subseteq \Lambda_1$ be the coset $(\Lambda_1 \cap \Lambda_2) + \mathbf{v}$ in Λ_1. Let $J_\mathbf{v} \subseteq \mathbb{Z}^n$ be the coset $\Lambda_2 + \mathbf{v}$ of Λ_2 in $\Lambda_1 + \Lambda_2$. Then

$$J_\mathbf{v} \cap \Lambda_1 = I_\mathbf{v}. \tag{3.1}$$

To see why this is true, first observe that all $\mathbf{u} \in I_\mathbf{v}$ are in Λ_1 and in $\Lambda_2 + \mathbf{v} = J_\mathbf{v}$ by definition, proving containment in one direction. Second, observe that since $\mathbf{v} \in \Lambda_1$, all $\mathbf{w} \in (\Lambda_2 + \mathbf{v}) \cap \Lambda_1$ satisfy $\mathbf{w} - \mathbf{v} \in \Lambda_1 \cap \Lambda_2$. Therefore $\mathbf{w} \in I_\mathbf{v}$, proving containment in the other direction, and (3.1) follows.

By definition we know that $\{I_\mathbf{v}\}_{\mathbf{v}\in\Lambda_1}$ contains all the cosets of $\Lambda_1 \cap \Lambda_2$ in Λ_1. Therefore, by the one-to-one correspondence, $\{J_\mathbf{v}\}_{\mathbf{v}\in\Lambda_1}$ must contain all the cosets of Λ_2 in $\Lambda_1 + \Lambda_2$. By (3.1), the distribution $\mathcal{D}_{\Lambda_1,\sigma,\mathbf{c}}$ assigns the same weight to $J_\mathbf{v}$ as to $I_\mathbf{v}$. But then, since $\mathcal{D}_{\Lambda_1,\sigma,\mathbf{c}}$ is statistically close to uniform over $\{I_\mathbf{v}\}_{\mathbf{v}\in\Lambda_1}$, it follows that $\mathcal{D}_{\Lambda_1,\sigma,\mathbf{c}}$ is also statistically close to uniform over $\{J_\mathbf{v}\}_{\mathbf{v}\in\Lambda_1}$, as required. □

Complexity assumption. We define a generalization of the now-standard *Small Integer Solution* (SIS) problem, which is to find a short nonzero vector in a certain class of lattices.

Definition 3.10. Let $\mathcal{L} = \{\mathcal{L}_n\}$ be a distribution ensemble of integer lattices, where lattices in $\mathcal{L}_n$ have dimension n. An instance of the $\mathcal{L}$-$\mathsf{SIS}_{n,\beta}$ problem is a lattice $\Lambda \leftarrow \mathcal{L}_n$. A solution to the problem is a nonzero vector $\mathbf{v} \in \Lambda$ with $\|\mathbf{v}\| \leq \beta$.

If $\mathcal{B}$ is an algorithm that takes as input a lattice Λ, we define the *advantage* of $\mathcal{B}$, denoted $\mathcal{L}$-SIS-$\mathrm{Adv}[\mathcal{B}, (n, \beta)]$, to be the probability that $\mathcal{B}$ outputs a solution to an $\mathcal{L}$-$\mathsf{SIS}_{n,\beta}$ problem instance Λ chosen according to the distribution $\mathcal{L}_n$.

We say that the $\mathcal{L}$-$\mathsf{SIS}_{n,\beta}$ problem is *infeasible* if for all polynomial-time algorithms $\mathcal{B}$, the advantage $\mathcal{L}$-SIS-$\mathrm{Adv}[\mathcal{B}, (n, \beta)]$ is a negligible function of n.

When $\mathcal{L}_n$ consists of $\Lambda_q^{\perp}(\mathbf{A})$ for uniformly random $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$, the $\mathcal{L}$-$\mathsf{SIS}_{n,\beta}$ problem is the standard $\mathsf{SIS}_{q,n,\beta}$ problem defined by Micciancio and Regev [32]. For this distribution of lattices, an algorithm that solves the $\mathcal{L}$-$\mathsf{SIS}_{n,\beta}$ problem can be used to solve worst-case problems on arbitrary ℓ-dimensional lattices [32, §5].

4 Homomorphic Signatures for Linear Functions over Small Fields

As a "warm-up" to our polynomially homomorphic scheme, we describe a signature scheme that can authenticate any linear function of signed vectors defined over small fields $\mathbb{F}_p$. Previous constructions can only achieve this functionality for vectors defined over large fields [12, 8] or for a small number of vectors [7]. In particular, our scheme easily accommodates binary data ($p = 2$). Linearly homomorphic signatures over $\mathbb{F}_2$ are an example of a primitive that can be built from lattices, but cannot currently be built from discrete-log or RSA-type assumptions (cf. the discussion in [7]). In Appendix A The instantiation of our abstract system for $\mathbb{F}_2$ is currently the best such system; see Section 4.2 for further details. we describe a variant of the scheme in which the data can take values in large fields $\mathbb{F}_p$.

Security is based on the SIS problem on q-ary lattices for some prime q; Micciancio and Regev [32], building on the work of Ajtai [2], show that this problem is as hard as standard worst-case problems on arbitrary lattices of dimension approximately $n/\lg q$. The system in this section is only secure for small p, specifically $p = \mathrm{poly}(n)$ with $p \leq \sqrt{q}/nk$ for data sets of size k.

Overview of the scheme. Since our system builds on the "hash-and-sign" signatures of Gentry, Peikert, and Vaikuntanathan [20], let us recall how GPV signatures work in an abstract sense. The public key is a lattice $\Lambda \subset \mathbb{Z}^n$ and the secret key is a short basis of Λ. To sign a message m, the secret key holder hashes m to an element $H(m) \in \mathbb{Z}^n/\Lambda$ and samples a short vector σ from the coset of Λ defined by $H(m)$. To verify σ, one checks that σ is short and that $\sigma \bmod \Lambda = H(m)$.

Recall that in a homomorphic signature scheme we wish to authenticate triples $(\tau, m, \langle f \rangle)$, where τ is a "tag" attached to a data set, m is a message in $\mathbb{F}_p^n$, and $\langle f \rangle$ is an encoding of a function f acting on k-tuples of messages. We encode a linear function $f : (\mathbb{F}_p^n)^k \to \mathbb{F}_p^n$ defined by $f(m_1, \ldots, m_k) = \sum_{i=1}^k c_i m_i$ by interpreting the c_i as integers in $(-p/2, p/2]$ and defining $\langle f \rangle := (c_1, \ldots, c_k) \in \mathbb{Z}^k$.

To authenticate both the message and the function as well as bind them together, we compute a single GPV signature that is *simultaneously* a signature on the (unhashed) message $m \in \mathbb{F}_p^n$ and a signature on a hash of $\langle f \rangle$.

This dual-role signature is computed via what we call the "intersection method," which works as follows. Let Λ_1 and Λ_2 be n-dimensional integer lattices with $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$. Suppose $m \in \mathbb{Z}^n/\Lambda_1$ is a message and ω_τ is a hash function (depending on the tag τ) that maps encodings of functions f to elements of $\mathbb{Z}^n/\Lambda_2$. Since the message m defines a coset of Λ_1 in $\mathbb{Z}^n$ and the hash $\omega_\tau(\langle f \rangle)$ defines a coset of Λ_2 in $\mathbb{Z}^n$, by the

Chinese remainder theorem the pair $\big(\ m,\ \omega_\tau(\langle f\rangle)\ \big)$ defines a unique coset of $\Lambda_1 \cap \Lambda_2$ in $\mathbb{Z}^n$. We can thus use a short basis of $\Lambda_1 \cap \Lambda_2$ to compute a short vector in this coset; i.e., a short vector σ with the property that $\sigma \bmod \Lambda_1 = m$ and $\sigma \bmod \Lambda_2 = \omega_\tau(\langle f\rangle)$. The vector σ is a signature on $(\tau, m, \langle f\rangle)$.

The $\mathsf{Sign}(\mathsf{sk}, \tau, m, i)$ algorithm uses the procedure above to generate a fresh signature on the triple $(\tau, m, \langle \pi_i\rangle)$ where π_i is the ith *projection function* defined by $\pi_i(m_1, \ldots, m_k) = m_i$ and encoded as $\langle \pi_i\rangle = \mathbf{e}_i$, the ith unit vector in $\mathbb{Z}^k$.

The homomorphic property is now obtained as follows. To authenticate the linear combination $m = \sum_{i=1}^k c_i m_i$ for integers c_i, we compute the signature $\sigma := \sum_{i=1}^k c_i \sigma_i$. If k and p are sufficiently small, then σ is a short vector. Furthermore, we have

$$\begin{aligned}\sigma \bmod \Lambda_1 &= \textstyle\sum_{i=1}^k c_i m_i = m\ , \quad \text{and}\\ \sigma \bmod \Lambda_2 &= \textstyle\sum_{i=1}^k c_i \omega_\tau(\langle \pi_i\rangle) = \sum_{i=1}^k c_i \omega_\tau(\mathbf{e}_i).\end{aligned}$$

Now suppose that ω_τ is linear, namely $\sum_{i=1}^k c_i \omega_\tau(\mathbf{e}_i) = \omega_\tau\big((c_1, \ldots, c_k)\big)$ for all $c_1, \ldots, c_k$ in $\mathbb{Z}$. Then since $(c_1, \ldots, c_k)$ is exactly the encoding of the function f defined by $f(m_1, \ldots, m_k) = \sum_{i=1}^k c_i m_i$, the signature σ authenticates both the message m and the fact that it was computed correctly (i.e., via f) from the original messages $m_1, \ldots, m_k$.

We now describe the construction concretely. Let p and q be distinct primes, let $\Lambda_1 = p\mathbb{Z}^n$, and let $\Lambda_2 = \Lambda_q^\perp(\mathbf{A})$ for some $\mathbf{A} \in \mathbb{F}_q^{\ell \times n}$. Messages are elements of $\mathbb{Z}^n/\Lambda_1 \cong \mathbb{F}_p^n$, with the isomorphism given explicitly by the map $\mathbf{x} \mapsto (\mathbf{x} \bmod \Lambda_1)$ that reduces the coefficients of $\mathbf{x}$ mod p. The hash function ω_τ takes values in $\mathbb{Z}^k$ and outputs elements in $\mathbb{Z}^n/\Lambda_2 \cong \mathbb{F}_q^\ell$, with the isomorphism given explicitly by the map $\mathbf{x} \mapsto (\mathbf{x} \bmod \Lambda_2)$ that sends $\mathbf{x}$ to $\mathbf{A} \cdot \mathbf{x} \bmod q$. Now given vectors $m \in \mathbb{F}_p^n$ and $\omega_\tau(\langle f\rangle) \in \mathbb{F}_q^\ell$, it is easy for anyone to compute an element $\mathbf{t} \in \mathbb{Z}^n$ of the corresponding coset of $\Lambda_1 \cap \Lambda_2$, namely

$$\mathbf{t} \bmod \Lambda_1 = m \quad \text{and} \quad \mathbf{t} \bmod \Lambda_2 = \omega_\tau(\langle f\rangle)\ .$$

If we have a short basis of $\Lambda_1 \cap \Lambda_2$, then we sign (τ, m, f) by using $\mathsf{SamplePre}$ to sample a short vector from the coset $(\Lambda_1 \cap \Lambda_2) + \mathbf{t}$.

Finally, we define the hash function ω_τ. Let $H\colon \{0,1\}^* \to (\mathbb{F}_q^\ell)$ be a hash function, and set $\alpha_i := H(\tau \| i)$ for $i = 1, \ldots, k$. Then for a function f encoded as $\langle f\rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, we define

$$\omega_\tau(\langle f\rangle) := \sum_{i=1}^k c_i \alpha_i \quad \in \mathbb{F}_q^\ell \cong \mathbb{Z}^n/\Lambda_2\ .$$

In particular, we have $\omega_\tau(\mathbf{e}_i) = \alpha_i$, and $\sum c_i \omega_\tau(\mathbf{e}_i) = \omega_\tau((c_1, \ldots, c_k))$, which is the property we needed to ensure that the scheme is homomorphic.

The linearly homomorphic scheme. We now describe the scheme.

$\mathsf{Setup}(1^n, k)$. On input a security parameter n and a maximum data set size k, do the following:

1. Choose two primes $p, q = \mathrm{poly}(n)$ with $q \geq (nkp)^2$. Define $\ell := \lfloor n/6 \log q \rfloor$.
2. Set $\Lambda_1 := p\mathbb{Z}^n$.
3. Use $\mathsf{TrapGen}(q, \ell, n)$ to generate a matrix $\mathbf{A} \in \mathbb{F}_q^{\ell \times n}$ along with a short basis $\mathbf{T}_q$ of $\Lambda_q^\perp(\mathbf{A})$. Define $\Lambda_2 := \Lambda_q^\perp(\mathbf{A})$ and $\mathbf{T} := p \cdot \mathbf{T}_q$. Note that $\mathbf{T}$ is a basis of $\Lambda_1 \cap \Lambda_2 = p\Lambda_2$.
4. Set $\nu := p \cdot \sqrt{n \log q} \cdot \log n$.
5. Let $H\colon \{0,1\}^* \to \mathbb{F}_q^\ell$ be a hash function (modeled as a random oracle).
6. Output the public key $\mathsf{pk} := (\Lambda_1, \Lambda_2, \nu, k, H)$ and the secret key $\mathsf{sk} = \mathbf{T}$.

The public key pk defines the following system parameters:

- The message space is $\mathbb{F}_p^n$ and signatures are short vectors in $\mathbb{Z}^n$.
- The set of admissible functions $\mathcal{F}$ is all $\mathbb{F}_p$-linear functions on k-tuples of messages in $\mathbb{F}_p^n$.
- For a function $f \in \mathcal{F}$ defined by $f(m_1, \ldots, m_k) = \sum_{i=1}^k c_i m_i$, we encode f by interpreting the c_i as integers in $(-p/2, p/2]$ and defining $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$.
- To evaluate the hash function ω_τ on an encoded function $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, do the following:
  (a) For $i = 1, \ldots, k$, compute $\alpha_i \leftarrow H(\tau \| i)$ in $\mathbb{F}_q^\ell$.
  (b) Define $\omega_\tau(\langle f \rangle) := \sum_{i=1}^k c_i \alpha_i \ \in \mathbb{F}_q^\ell$.

$\mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. On input a secret key sk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_p^n$, and an index i, do:

1. Compute $\alpha_i := H(\tau \| i) \in \mathbb{F}_q^\ell$. Then, by definition, $\omega_\tau(\langle \pi_i \rangle) = \alpha_i$.
2. Compute $\mathbf{t} \in \mathbb{Z}^n$ such that $\mathbf{t} \bmod p = m$ and $\mathbf{A} \cdot \mathbf{t} \bmod q = \alpha_i$.
3. Output $\sigma \leftarrow \mathsf{SamplePre}(\Lambda_1 \cap \Lambda_2, \mathbf{T}, \mathbf{t}, \nu) \ \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$.

$\mathsf{Verify}(\mathsf{pk}, \tau, m, \sigma, f)$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_p^n$, a signature $\sigma \in \mathbb{Z}^n$, and a function $f \in \mathcal{F}$, do:

1. If all of the following conditions hold, output 1 (accept); otherwise output 0 (reject):
   (a) $\|\sigma\| \le k \cdot \frac{p}{2} \cdot \nu\sqrt{n}$.
   (b) $\sigma \bmod p = m$.
   (c) $\mathbf{A} \cdot \sigma \bmod q = \omega_\tau(\langle f \rangle)$.

$\mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma})$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a function $f \in \mathcal{F}$ encoded as $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, and a tuple of signatures $\sigma_1, \ldots, \sigma_k \in \mathbb{Z}^n$, output $\sigma := \sum_{i=1}^k c_i \sigma_i$.

Lemma 4.1. *The linearly homomorphic signature scheme defined above is correct with overwhelming probability.*

Proof. Let $\tau \in \{0,1\}^n$, $m \in \mathbb{F}_p^n$, and $i \in \{1, \ldots, k\}$, and set $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. We check the three verification conditions relative to the projection function π_i defined by $\pi_i(m_1, \ldots, m_k) = m_i$ and encoded as $\langle \pi_i \rangle = \mathbf{e}_i$.

(a) By Theorem 3.2, we have $\|\widetilde{\mathbf{T}}\| \le O(p \cdot \sqrt{\ell \log q})$, and therefore $\nu \ge \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log \ell})$. By Lemma 3.6, we have $\|\sigma\| \le \nu\sqrt{n}$ with overwhelming probability.

(b) By correctness of $\mathsf{SamplePre}$, we have $\sigma \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$, and thus $\sigma \bmod p = \mathbf{t} \bmod p$. By definition of $\mathbf{t}$, we have $\mathbf{t} \bmod p = m$.

(c) As above, we have $\sigma \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$. Since $\Lambda_2 = \Lambda_q^\perp(\mathbf{A})$, we have $\mathbf{A} \cdot \sigma \bmod q = \mathbf{A} \cdot \mathbf{t} \bmod q = \alpha_i$. By our definition of ω_τ, we have $\omega_\tau(\langle \pi_i \rangle) = \alpha_i$.

Now let $\tau \in \{0,1\}^n$, $\vec{m} = (m_1, \ldots, m_k) \in (\mathbb{F}_p^n)^k$, and $f \in \mathcal{F}$ encoded as $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$. Let $\vec{\sigma} = (\sigma_1, \ldots, \sigma_k)$ with $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m_i, i)$. Then by our definition of $\mathsf{Evaluate}$, we have

$$\sigma' := \mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma}) = \textstyle\sum_i c_i \sigma_i$$

We check the three verification conditions for the signature σ' on the message $f(\vec{m})$, relative to the function f.

(a) Since $|c_i| \le p/2$ for all i, we have that with overwhelming probability,

$$\|\sigma'\| \ \le \ k \cdot \frac{p}{2} \cdot \max\{\|\sigma_i\| : \sigma_i \in \vec{\sigma}\} \ \le \ k \cdot \frac{p}{2} \cdot \nu\sqrt{n},$$

where the second inequality follows from Lemma 3.6.

(b) By correctness of individual signatures, we have $\sigma_i \bmod p = m_i$ for $i \in \{1, \ldots, k\}$. We thus have

$$\sigma' \bmod p = \sum_i c_i \sigma_i \bmod p = \sum_i c_i m_i = f(\vec{m}).$$

(c) By correctness of individual signatures, we have $\mathbf{A} \cdot \sigma_i \bmod q = \alpha_i$ for $i \in \{1, \ldots, k\}$. It follows that

$$\mathbf{A} \cdot \sigma' \bmod q = \sum_i \mathbf{A} \cdot c_i \sigma_i \bmod q = \sum_i c_i \alpha_i \bmod q = \omega_\tau(\langle f \rangle). \qquad \square$$

4.1 Length efficiency

With overwhelming probability, signatures output by $\mathsf{Sign}(\mathsf{sk}, \cdot, \cdot, \cdot)$ have norm at most $\nu\sqrt{n}$ and therefore bit length at most $n \lg(\nu\sqrt{n})$. As we saw in the proof of Lemma 4.1, applying Evaluate to a linear function $f : (\mathbb{F}_p^n)^k \to \mathbb{F}_p^n$ and k signatures generated by Sign outputs a signature σ whose norm is at most $k \cdot \frac{p}{2} \cdot \nu\sqrt{n}$ with high probability. Therefore, the bit length of the derived signature σ satisfies

$$\mathrm{len}(\sigma) \leq n \lg k + n \lg \left(\frac{p}{2} \cdot \nu\sqrt{n} \right).$$

Since the bit length of the derived signature depends logarithmically on k, we conclude that the scheme is length efficient.

4.2 Unforgeability

We will show that an adversary that forges a signature for the linearly homomorphic scheme can be used to compute a short vector in the lattice Λ_2 chosen in Step 3 of Setup. By Theorem 3.2, the distribution of matrices $\mathbf{A}$ used to define Λ_2 is statistically close to uniform over $\mathbb{F}_q^{\ell \times n}$. Thus the distribution of lattices Λ_2 output by Setup is statistically close to the distribution of challenges for the $\mathsf{SIS}_{q,n,\beta}$ problem (for any β). Our security theorem is as follows.

Theorem 4.2. *If $\mathsf{SIS}_{q,n,\beta}$ is infeasible for $\beta = k \cdot p^2 \cdot n \log n \sqrt{\log q}$, then the linearly homomorphic signature scheme above is unforgeable in the random oracle model.*

The full proof of this theorem is given in Section 7.3. In particular, Theorem 4.2 is a special case of Theorem 7.4, which proves unforgeability of an abstract homomorphic signature system. Here we sketch the intuition behind the proof without the formalism of the abstract system.

Sketch of Proof. Let $\mathcal{A}$ be an adversary that plays the security game of Definition 2.1. Given a challenge lattice $\Lambda_2 := \Lambda_q^\perp(\mathbf{A})$ for $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$, we simulate the Sign algorithm on input (τ, m, i) for random τ by sampling a short vector σ from $\mathcal{D}_{\Lambda_1 + m, \nu}$ and defining $H(\tau \| i) := \mathbf{A} \cdot \sigma \bmod q$. Then σ is a valid signature on (τ, m, i). Other queries to H are answered similarly, but with a random message m. By Lemma 3.9, the Gaussian parameter ν is large enough so that $H(\tau \| i)$ is statistically close to uniform in $\mathbb{F}_q^\ell$.

Eventually $\mathcal{A}$ outputs a tag τ^*, a message m^*, a function f encoded as $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, and a signature σ^*. Let σ_i be the short vector chosen when programming $H(\tau^* \| i)$, and let $\sigma_f := \sum_i c_i \sigma_i$. We claim that if $\mathcal{A}$ outputs a valid forgery, then with high probability the vector $\sigma^* - \sigma_f$ is a nonzero vector in Λ_2 of length at most β; i.e., a solution to the $\mathsf{SIS}_{q,n,\beta}$ problem.

Suppose $\mathcal{A}$ outputs a type 2 forgery, so the simulator has generated signatures $\vec{\sigma} = (\sigma_1, \ldots, \sigma_k)$ on messages $\vec{m} = (m_1, \ldots, m_k)$ using the tag τ^*. First observe that the verification condition (1a) implies that $\|\sigma^*\|$ and $\|\sigma_f\|$ are both less than $k \cdot \frac{p}{2} \cdot \nu\sqrt{n}$, and therefore $\|\sigma^* - \sigma_f\| \leq \beta$. Next observe that if the forgery is

valid, then $m^* \neq f(\vec{m})$. The verification condition (1b) implies that $(\sigma^* - \sigma_f) \bmod p = m^* - f(\vec{m}) \neq 0$, and thus $\sigma^* - \sigma_f \neq 0$. On the other hand, verification condition (1c) implies that $\mathbf{A} \cdot \sigma^* \bmod q = \mathbf{A} \cdot \sigma_f \bmod q$, and thus $\sigma^* - \sigma_f \in \Lambda_2$.

The argument for a type 1 forgery is similar, using random messages $\vec{m}$ instead of queried ones. □

Worst-case connections. By [20, Proposition 5.7], if $q \geq \beta \cdot \omega(\sqrt{n \log n})$, then the $\mathsf{SIS}_{q,m,\beta}$ problem is as hard as approximating the SIVP problem in the worst case to within $\beta \cdot \tilde{O}(\sqrt{n})$ factors. Our requirement in the Setup algorithm that $q \geq (nkp)^2$ guarantees that q is sufficiently large for this theorem to apply. While the exact worst-case approximation factor will depend on the parameters k and p, it is polynomial in n in any case.

Comparison with prior work. Boneh and Freeman [7] describe a linearly homomorphic signature scheme that can authenticate vectors over $\mathbb{F}_p$ for small p, with unforgeability also depending on the SIS problem. However, for their system to securely sign k messages, the $\mathsf{SIS}_{q,2n-k,\beta}$ problem must be difficult for $\beta = \tilde{O}(k^{3/2} \cdot k! \cdot (n/\lg q)^{k/2+1})$, and therefore their system is designed to only sign a constant number of vectors per data set ($k = O(1)$) while maintaining a polynomial connection to worst-case lattice problems. On the other hand, for the same value of q our system remains secure when signing $k = \mathrm{poly}(n)$ vectors per data set.

4.3 Privacy

We now show that our linearly homomorphic signature scheme is weakly context hiding. Specifically, we use Theorem 3.7 to show that a derived signature on a linear combination $m' = \sum_{i=1}^{k} c_i m_i$ depends (up to negligible statistical distance) only on m' and the c_i, and not on the initial messages m_i. Consequently, even an unbounded adversary cannot win the privacy game of Definition 2.3. To prove privacy when multiple derived signatures are revealed we need the generalization of Theorem 3.7 given in Theorem 3.8.

Theorem 4.3. *Suppose that ν defined in the* Setup *algorithm satisfies $\nu > p^{s+1} \cdot k^s \cdot \omega(\sqrt{\log n})$. Then the linearly homomorphic signature scheme described above is s-weakly context hiding for data sets of size at most k.*

Proof. We follow the argument of [7, Theorem 5.4], which is an unconditional argument for weak context hiding. In the privacy game, let $(\vec{m}_0^*, \vec{m}_1^*, f_1, \ldots, f_s)$ be the adversary's output in the challenge phase, where $\vec{m}_0^*$ and $\vec{m}_1^*$ contain k messages each. For convenience we treat $\vec{m}_b^*$ as a matrix in $\mathbb{F}_p^{k \times n}$ whose jth row is message number j in $\vec{m}_b^*$. We know that

$$m_i := f(\vec{m}_0^*) = f(\vec{m}_1^*) \in \mathbb{F}_p^n \qquad \text{for all } i = 1, \ldots, s. \tag{4.1}$$

Let $\Lambda := \Lambda_1 \cap \Lambda_2$ and let τ be the tag used to answer the challenge. For $j = 1, \ldots, k$, let $\sigma_j^{(0)}$ and $\sigma_j^{(1)}$ be the challenger's signatures (in $\mathbb{Z}^n$) on the jth vector in $\vec{m}_0^*$ and $\vec{m}_1^*$, respectively. We arrange these signatures in two matrices $E^{(0)}, E^{(1)} \in \mathbb{Z}^{k \times n}$ so that row j of $E^{(b)}$ is $(\sigma_j^{(b)})^\intercal \in \mathbb{Z}^n$.

For $i = 1, \ldots, s$ let $\mathbf{d}_i^{(b)}$ be a derived signature on m_i computed using the Evaluate algorithm applied to the signatures $\{\sigma_j^{(b)}\}_{1 \leq j \leq k}$ and the function f_i. Again, we arrange these derived signatures in two matrices $D^{(0)}, D^{(1)} \in \mathbb{Z}^{s \times n}$ so that row i of $D^{(b)}$ is $(\mathbf{d}_i^{(b)})^\intercal \in \mathbb{Z}^n$. The challenger chooses a bit b and gives the adversary the signatures $D^{(b)}$. We show that the derived signatures $D^{(0)}$ and $D^{(1)}$ are sampled from statistically close distributions so that the adversary cannot guess b with non-negligible advantage.

By definition of algorithm Evaluate, there is a matrix $F \in \mathbb{Z}^{s\times k}$ such that $D^{(b)} = FE^{(b)}$ for $b = 0, 1$. The ith row of F is determined by the function f_i and is equal to $\omega_\tau(\langle f_i \rangle)$; in particular, all the entries of F are in $(-p/2, p/2]$. By (4.1) we know that $F\vec{m}_0^* = F\vec{m}_1^*$ (where we identify $\mathbb{Z} \cap (-p/2, p/2]$ with $\mathbb{F}_p$).

Suppose $b = 0$. By definition of algorithm Sign, every signature $\sigma_j^{(0)}$ is sampled from a distribution statistically close to $\mathcal{D}_{\Lambda+\mathbf{t}_j,\nu}$ where $\mathbf{t}_j$ is the vector computed in Step (2) of the Sign algorithm, namely $\mathbf{t}_j \bmod \Lambda_1 = m_{0,j}^*$ and $\mathbf{t}_j \bmod \Lambda_2 = \omega_\tau(\langle \pi_j \rangle)$. All of these signatures are mutually independent and therefore, by Theorem 3.8, the derived signatures $D^{(0)} = FE^{(0)}$ are statistically close to a distribution parametrized by Λ, σ, F, and $F\vec{m}_0^*$. (Note that the rows of $\vec{m}_0^*$ describe cosets of Λ so that $\vec{m}_0^*$ plays the same role as the matrix T in Theorem 3.8.) Since the same holds for $b = 1$ and since $F\vec{m}_0^* = F\vec{m}_1^*$, we see that $D^{(0)}$ and $D^{(1)}$ are statistically close. Consequently, even an unbounded adversary cannot win the privacy game with non-negligible advantage.

It remains to argue that ν is sufficiently large to apply Theorem 3.8 to $FE^{(b)}$, namely that $\nu > \mu(F)\eta_\epsilon(\Lambda)$ for some negligible ϵ, where $\mu(F)$ is defined as in Theorem 3.8. First, recall that the determinant of a matrix is less than the product of the norms of its rows. Therefore, since the entries of F are in $(-p/2, p/2]$, it is not difficult to see that $\mu(F) \le (pk)^s$. Second, since $\Lambda = \Lambda_1 \cap \Lambda_2 = p\mathbb{Z}^n \cap \Lambda_2 = p\Lambda_2$, Lemma 3.5 shows that with overwhelming probability we have $\eta_\epsilon(\Lambda) = p \cdot \eta_\epsilon(\Lambda_2) < p \cdot \omega(\sqrt{\log n})$ for negligible ϵ. Combining these two bounds and the bound on ν, we obtain

$$\mu(F)\eta_\epsilon(\Lambda) \le p \cdot (pk)^s \cdot \omega(\sqrt{\log n}) < \nu$$

as required for Theorem 3.8. □

Remark 4.4. The exponential dependence on the number s of allowed function queries in Theorem 4.3 shows that for the parameter ν defined in the Setup algorithm, the scheme is private only when a small number of derived signatures are revealed. If instead we require Setup to use a ν such that $\nu > p^{s+1} \cdot k^s \cdot \omega(\sqrt{\log n})$, then the scheme is weakly context hiding when an arbitrary number of derived signatures are revealed. Of course, making ν so large forces us to increase the parameter β in Theorem 4.2, thus weakening the unforgeability property.

In addition, we observe that when $s > k$, the result of Theorem 4.3 holds whenever $\nu > p \cdot (pk)^k \cdot \omega(\sqrt{\log n})$. In particular, for small data sets the scheme is weakly context hiding: it remains private even against an adversary allowed to make an arbitrary number of function queries.

5 Background on Ideal Lattices

A *number field* is a finite-degree algebraic extension of the rational numbers $\mathbb{Q}$. Any number field K can be represented as $\mathbb{Q}[x]/(f(x))$ for some monic, irreducible polynomial $f(x)$ with integer coefficients (and for each K there are infinitely many such f). The *degree* of a number field K is its dimension as a vector space over $\mathbb{Q}$, and is also the degree of any polynomial f defining K. For any given f, the set $\{1, x, x^2, \ldots, x^{\deg f-1}\}$ is a $\mathbb{Q}$-basis for K, and we can therefore identify K with $\mathbb{Q}^n$ by mapping a polynomial of degree less than n to its vector of coefficients. By identifying K with $\mathbb{Q}^n$ using this "coefficient embedding," we can define a length function $\|\cdot\|$ on elements of K simply by using any norm on $\mathbb{Q}^n$. This length function is non-canonical — it depends explicitly on the choice of f used to represent K. (Here all norms will be the ℓ_2 norm unless otherwise stated.)

Our identification of $K = \mathbb{Q}[x]/(f(x))$ with $\mathbb{Q}^n$ induces a *multiplicative* structure on $\mathbb{Q}^n$ in addition to the usual *additive* structure. We define a parameter

$$\gamma_f := \sup_{u,v\in K} \frac{\|u \cdot v\|}{\|u\| \cdot \|v\|}.$$

This parameter bounds how much multiplication can increase the length of the product, relative to the product of the length of the factors. For our applications we will need to have $\gamma_f = \text{poly}(n)$. If n is a power of 2, then the function $f(x) = x^n + 1$ has $\gamma_f \leq \sqrt{n}$ (cf. [18, Lemma 7.4.3]). We will think of this $f(x)$ as our "preferred" choice for applications.

Number rings and ideals. A *number ring* is a ring whose field of fractions is a number field K. A survey of arithmetic in number rings can be found in [43]; here we summarize the key points.

Every number field has a subring, called the *ring of integers* and denoted by $\mathcal{O}_K$, that plays the same role with respect to K as the integers $\mathbb{Z}$ do with respect to $\mathbb{Q}$. The ring of integers consists of all elements of K whose characteristic polynomials have integer coefficients. Under the identification of K with $\mathbb{Q}^n$, the ring $\mathcal{O}_K$ forms a full-rank discrete subgroup of $\mathbb{Q}^n$; i.e., a lattice. Inside $\mathcal{O}_K$ is the subring $R = \mathbb{Z}[x]/(f(x))$. Under our identification of K with $\mathbb{Q}^n$, the ring R corresponds to $\mathbb{Z}^n$. In general R is a proper sublattice of $\mathcal{O}_K$. The "distance" between R and $\mathcal{O}_K$ is measured by the *conductor* $\mathfrak{f}_R$, which is the largest ideal of $\mathcal{O}_K$ contained in R.

An *(integral) ideal* of R is an additive subgroup $I \subset R$ that is closed under multiplication by elements of R. By our identification of R with $\mathbb{Z}^n$, the ideal I is a sublattice of R and is therefore also called an *ideal lattice*. Note that this usage of "ideal lattice" to refer to a rank one R-module differs from that of [41, 27], which use the terminology to refer to R-modules of arbitrary rank.

An ideal $I \subset R$ is *prime* if for $x, y \in R$, $xy \in I$ implies either $x \in I$ or $y \in I$. If $\mathfrak{p}$ is a prime ideal, then $R/\mathfrak{p}$ is a finite field $\mathbb{F}_{p^e}$; the integer e is the *degree* of $\mathfrak{p}$ and the prime p is the *characteristic* of $\mathfrak{p}$. An ideal I is *principal* if it can be written as $\alpha \cdot R$ for some $\alpha \in R$. In general most ideals are not principal; the proportion of principal ideals is 1 over the size of the *class group* of R, which is exponential in n. The *norm* of an ideal I is the size of the (additive) group R/I.

If $\mathfrak{p}$ is a prime ideal of R, then by a theorem of Kummer and Dedekind [43, Theorem 8.2] we can write $\mathfrak{p} = p \cdot R + h(x) \cdot R$ for some polynomial $h(x)$ whose reduction mod p is an irreducible factor of $f(x) \bmod p$. Writing $\mathfrak{p}$ in this "two-element representation" makes it easy to compute the corresponding quotient map $\mathbb{Z}[x]/(f(x)) \to \mathbb{F}_{p^e}$; we simply reduce a polynomial in $\mathbb{Z}[x]$ modulo both p and $h(x)$. In particular, if $\mathfrak{p}$ is a degree-one prime, then $h(x) = x - \alpha$ for some integer α and the quotient map is given by $z(x) \mapsto z(\alpha) \bmod p$.

Generating ideals with a short basis. If we are to use ideals as the lattices Λ_1 and Λ_2 in our abstract signature scheme, we will need a method for generating ideals $\mathfrak{p}$ and $\mathfrak{q}$ in R along with a short basis for $\mathfrak{p} \cap \mathfrak{q}$ (which is equal to $\mathfrak{p} \cdot \mathfrak{q}$ if $\mathfrak{p}$ and $\mathfrak{q}$ are relatively prime ideals). Furthermore, our security proof requires that given $\mathfrak{q}$ *without* a short basis, we can still compute a prime $\mathfrak{p}$ with a short basis.

In our construction we generate ideals using an algorithm of Smart and Vercauteren [39]. This algorithm generates a *principal* prime ideal $\mathfrak{p}$ along with a short generator g of $\mathfrak{p}$. We can multiply g by powers of x to generate a full-rank set of vectors $\{g, xg, x^2g, \ldots, x^{n-1}g\}$ that spans $\mathfrak{p}$. Since $\|x\| = 1$, we have $\|x^i g\| \leq \gamma_f \cdot \|g\|$, so if γ_f is small then these vectors are all short.

Theorem 5.1 ([39, §3.1])**.** *There is an algorithm* PrincGen *that takes input a monic irreducible polynomial $f(x) \in \mathbb{Z}[x]$ of degree n and a parameter δ, and outputs a principal degree-one prime ideal $\mathfrak{p} = (p, x - a)$ in $K := \mathbb{Q}[x]/(f(x))$, along with a generator g of $\mathfrak{p}$ satisfying $\|g\| \leq \delta\sqrt{n}$.*

The algorithm works by sampling a random g with low norm and seeing if it generates a prime ideal in $\mathcal{O}_K$. Smart and Vercauteren do not give a rigorous analysis of the algorithm's running time, but heuristically we expect that by the number field analogue of the Prime Number Theorem [35, Theorem 8.9], we will find a prime ideal after trying $O(n \log n \log \delta)$ values of g.

6 Homomorphic Signatures for Polynomial Functions

In this section we describe our main construction, a signature scheme that authenticates polynomial functions on signed messages.

Recall the basic idea of our linearly homomorphic scheme from Section 4: messages are elements of $\mathbb{Z}^n \bmod \Lambda_1$, functions are mapped (via the hash function ω_τ) to elements of $\mathbb{Z}^n \bmod \Lambda_2$, and a signature on $(\tau, m, \langle f \rangle)$ is a short vector in the coset of $\Lambda_1 \cap \Lambda_2$ defined by m and $\omega_\tau(\langle f \rangle)$. To verify a signature σ, we simply confirm that σ is a short vector and that $\sigma \bmod \Lambda_1 = m$ and $\sigma \bmod \Lambda_2 = \omega_\tau(\langle f \rangle)$. The homomorphic property follows from the fact that the maps $\mathbf{x} \mapsto (\mathbf{x} \bmod \Lambda_i)$ are linear maps — i.e., *vector space homomorphisms* — and therefore adding signatures corresponds to adding the corresponding messages and (encoded) functions.

Our polynomial system is based on the following idea: what if the lattice $\mathbb{Z}^n$ has a *ring* structure and the lattices Λ_1, Λ_2 are *ideals*? Then the maps $\mathbf{x} \mapsto (\mathbf{x} \bmod \Lambda_i)$ are *ring homomorphisms*, and therefore adding *or multiplying* signatures corresponds to adding *or multiplying* the corresponding messages and functions. Since any polynomial can be computed by repeated additions and multiplications, adding this structure to our lattices allows us to authenticate polynomial functions on messages.

Concretely, we let $F(x) \in \mathbb{Z}[x]$ be a monic, irreducible polynomial of degree n. We define the number field $K = \mathbb{Q}[x]/(F(x))$ and identify K with $\mathbb{Q}^n$ via the coefficient embedding. Under this embedding, $R = \mathbb{Z}[x]/(F(x))$ corresponds to the lattice $\mathbb{Z}^n$. We now let Λ_1 and Λ_2 be (degree one) prime ideals $\mathfrak{p}, \mathfrak{q} \subset R$ of norm p, q respectively. We fix an isomorphism from $R/\mathfrak{p}$ to $\mathbb{F}_p$ by representing $\mathfrak{p}$ as $p \cdot R + (x - a) \cdot R$ and mapping $h(x) \in R$ to $h(a) \bmod p \in \mathbb{F}_p$, and similarly for $R/\mathfrak{q} \cong \mathbb{F}_q$. We can now sign messages exactly as in the linearly homomorphic scheme.

In our linearly homomorphic scheme we used the projection functions π_i as a generating set for admissible functions, and we encoded the function $f = \sum c_i \pi_i$ by its coefficient vector $(c_1, \ldots, c_k)$ (with the c_i interpreted as integers in $(-p/2, p/2]$). When we consider polynomial functions on $\mathbb{F}_p[x_1, \ldots, x_k]$, the projection functions π_i are exactly the linear monomials x_i, and we can obtain any (non-constant) polynomial function by adding and multiplying monomials. If we fix an ordering on all monomials of the form $x_1^{e_1} \cdots x_k^{e_k}$, then we can encode any polynomial function as its vector of coefficients, with the unit vectors $\mathbf{e}_i$ representing the linear monomials x_i for $i = 1, \ldots, k$.

The hash function ω_τ is defined exactly as in our linear scheme: for a function f in $\mathbb{F}_p[x_1, \ldots, x_k]$ whose encoding is $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathbb{Z}^\ell$, we define a polynomial $\hat{f} \in \mathbb{Z}[x_1, \ldots, x_k]$ that reduces to f mod p. We then define $\omega_\tau(\langle f \rangle) = \hat{f}(\alpha_1, \ldots, \alpha_k)$, where $\alpha_i \in \mathbb{F}_q$ are defined to be $H(\tau, i)$ for some hash function H.

We use the same lifting of f to $\hat{f} \in \mathbb{Z}[x_1, \ldots, x_k]$ to evaluate polynomials on signatures; specifically, given a polynomial f and signatures $\sigma_1, \ldots, \sigma_k \in K$ on messages $m_1, \ldots, m_k \in \mathbb{F}_p$, the signature on $f(m_1, \ldots, m_k)$ is given by $\hat{f}(\sigma_1, \ldots, \sigma_k)$.

Recall that for $\mathbf{v}_1, \mathbf{v}_2 \in R$, the length of $\mathbf{v}_1 \cdot \mathbf{v}_2$ is bounded by $\gamma_F \cdot \|\mathbf{v}_1\| \cdot \|\mathbf{v}_2\|$. Thus if we choose $F(x)$ so that γ_F is polynomial in n, then multiplying together a constant number of vectors of length $\mathrm{poly}(n)$ produces a vector of length $\mathrm{poly}(n)$. It follows that the derived signature $f(\sigma_1, \ldots, \sigma_k)$ is short as long as the degree of f is bounded and the coefficients of f are small (when lifted to the integers). The system therefore can support polynomial computations on messages for polynomials with small coefficients and bounded degree.

The polynomially homomorphic scheme. We now describe the scheme formally.

$\mathsf{Setup}(1^n, k)$. On input a security parameter n and a maximum data set size k, do the following:

1. Choose a monic irreducible polynomial $F(x) \in \mathbb{Z}[x]$ of degree n with $\gamma_F = \mathrm{poly}(n)$.
   Let $K := \mathbb{Q}[x]/(F(x))$ be embedded in $\mathbb{Q}^n$ via the coefficient embedding.

Let $R = \mathbb{Z}^n$ be the lattice corresponding $\mathbb{Z}[x]/(F(x)) \subset \mathcal{O}_K$.

2. Run the PrincGen algorithm twice on inputs F, n to produce distinct principal degree-one prime ideals $\mathfrak{p} = (p, x - a)$ and $\mathfrak{q} = (q, x - b)$ of R with generators $g_\mathfrak{p}, g_\mathfrak{q}$, respectively.
3. Let $\mathbf{T}$ be the basis $\{g_\mathfrak{p} g_\mathfrak{q}, g_\mathfrak{p} g_\mathfrak{q} x, \ldots, g_\mathfrak{p} g_\mathfrak{q} x^{n-1}\}$ of $\mathfrak{p} \cdot \mathfrak{q}$.
4. Define $\nu := \gamma_F^2 \cdot n^3 \log n$. Choose integers $y = \mathrm{poly}(n)$ and $d = O(1)$.
5. Let $H \colon \{0,1\}^* \to \mathbb{F}_q$ be a hash function (modeled as a random oracle).
6. Output the public key $\mathsf{pk} = (F, p, q, a, b, \nu, y, d, H)$ and secret key $\mathsf{sk} = \mathbf{T}$.

The public key pk defines the following system parameters:

- The message space is $\mathbb{F}_p$ and signatures are short vectors in R.
- The set of admissible functions $\mathcal{F}$ is all polynomials in $\mathbb{F}_p[x_1, \ldots, x_k]$ with coefficients in $\{-y, \ldots, y\}$, degree at most d, and constant term zero. The quantity y is only used in algorithm Verify.
- Let $\ell = \binom{k+d}{d} - 1$. Let $\{Y_j\}_{j=1}^{\ell}$ be the set of all non-constant monomials $x_1^{e_1} \cdots x_k^{e_k}$ of degree $\sum e_i \le d$, ordered lexicographically.[4] Then any polynomial function $f \in \mathcal{F}$ is defined by $f(\vec{m}) = \sum_{j=1}^{\ell} c_j Y_j(\vec{m})$ for $c_j \in \mathbb{F}_p$. We interpret the c_j as integers in $[-y, y]$ and encode f as $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathbb{Z}^\ell$.
- To evaluate the hash function ω_τ on an encoded function $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathbb{Z}^\ell$, do the following:
  (a) For $i = 1, \ldots, k$, compute $\alpha_i \leftarrow H(\tau \| i)$.
  (b) Define $\omega_\tau(\langle f \rangle) := \sum_{j=1}^{\ell} c_j Y_j(\alpha_1, \ldots, \alpha_k) \in \mathbb{F}_q$.

$\mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. On input a secret key sk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_p$, and an index i, do:

1. Compute $\alpha_i := H(\tau \| i) \in \mathbb{F}_q$.
2. Compute $h = h(x) \in R$ such that $h(a) \bmod p = m$ and $h(b) \bmod q = \alpha_i$.
3. Output $\sigma \leftarrow \mathsf{SamplePre}(\mathfrak{p} \cdot \mathfrak{q}, \mathbf{T}, h, \nu) \ \in (\mathfrak{p} \cdot \mathfrak{q}) + h$.

$\mathsf{Verify}(\mathsf{pk}, \tau, m, \sigma, f)$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_p$, a signature $\sigma = \sigma(x) \in R$, and a function $f \in \mathcal{F}$, do:

1. If all of the following conditions hold, output 1 (accept); otherwise output 0 (reject):
   (a) $\|\sigma\| \le \ell \cdot y \cdot \gamma_F^{d-1} \cdot (\nu\sqrt{n})^d$.
   (b) $\sigma(a) \bmod p = m$.
   (c) $\sigma(b) \bmod q = \omega_\tau(\langle f \rangle)$.

$\mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma})$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a function $f \in \mathcal{F}$ encoded as $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathbb{Z}^\ell$, and a tuple of signatures $\sigma_1, \ldots, \sigma_k \in \mathbb{Z}^n$, do:

1. Lift $f \in \mathbb{F}_p[x_1, \ldots, x_k]$ to $\mathbb{Z}[x_1, \ldots, x_k]$ by setting $\hat{f} := \sum_{j=1}^{\ell} c_j Y_j(x_1, \ldots, x_k)$.
2. Output $\hat{f}(\sigma_1, \ldots, \sigma_k)$.

Lemma 6.1. *The polynomially homomorphic signature scheme defined above is correct with overwhelming probability.*

Proof. Let $\tau \in \{0,1\}^n$, $m \in \mathbb{F}_p$, and $i \in \{1, \ldots, k\}$, and set $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. We check the three verification conditions relative to the projection function π_i, which in this context is the monomial x_i. (Note that $\langle x_i \rangle = \mathbf{e}_i$.)

(a) By Theorem 5.1 the generators $g_\mathfrak{p}, g_\mathfrak{q}$ have norm at most $n^{1.5}$, and therefore $\|g_\mathfrak{p} g_\mathfrak{q} x^i\| \le \gamma_F^2 \cdot n^3$ for $i = 0, \ldots, n-1$. It follows from Lemma 3.1 that the basis $\mathbf{T}$ has $\|\widetilde{\mathbf{T}}\| \le \gamma_F^2 n^3$, and therefore $\nu \ge \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$. By Lemma 3.6, we have $\|\sigma\| \le \nu\sqrt{n}$ with overwhelming probability.

[4] Specifically, we identify a monomial $x_1^{e_1} \cdots x_k^{e_k}$ with the non-decreasing sequence $(0, \ldots, 0, 1, \ldots, 1, \ldots\ldots, k, \ldots, k)$ of length d that has the integer i appearing e_i times (and 0 appearing $d - \sum e_i$ times), and use the lexicographic ordering on $\mathbb{Z}^d$. In particular, under this ordering all monomials of degree e appear before any monomial of degree $e + 1$.

(b) By correctness of SamplePre, we have $\sigma \in \mathfrak{p} \cdot \mathfrak{q} + h(x)$, and thus $\sigma(a) \bmod p = h(a) \bmod p$. By definition of $h(x)$, we have $h(a) \bmod p = m$.

(c) As above, we have $\sigma \in \mathfrak{p} \cdot \mathfrak{q} + h(x)$ and therefore $\sigma(b) \bmod q = h(b) \bmod q = \alpha_i$.

Now let $\tau \in \{0,1\}^n$, $\vec{m} = (m_1, \ldots, m_k) \in \mathbb{F}_p^k$, and $f \in \mathcal{F}$ encoded as $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathbb{Z}^\ell$. Let $\vec{\sigma} = (\sigma_1, \ldots, \sigma_k)$ with $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m_i, i)$. Interpret the c_j as integers in $[-y, y]$. Then by our definition of Evaluate, we have

$$\sigma' := \mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma}) = \textstyle\sum_j c_j Y_j(\vec{\sigma})$$

(Recall that $\{Y_j\}$ is the set of non-constant monomials $x_1^{e_i} \cdots x_k^{e_k}$ of degree at most d.) We check the three verification conditions for the signature σ' on the message $f(\vec{m})$, relative to the function f.

(a) Since evaluating $Y_j(\sigma)$ consists of multiplying together at most d of the σ_i, we have

$$\|Y_j(\vec{\sigma})\| \leq \gamma_F^{d-1} \cdot \left(\max\{\|\sigma_i\| : \sigma_i \in \vec{\sigma}\}\right)^d.$$

Since f is admissible we have $|c_j| \leq y$ for all j. Thus we have

$$\|\sigma'\| \;\leq\; \ell \cdot y \cdot \gamma_F^{d-1} \cdot \left(\max\{\|\sigma_i\| : \sigma_i \in \vec{\sigma}\}\right)^d \;\leq\; \ell \cdot y \cdot \gamma_F^{d-1} \cdot (\nu\sqrt{n})^d,$$

where the second inequality follows from Lemma 3.6.

(b) The fact that the map from R to $\mathbb{F}_p$ given by $h(x) \mapsto h(a) \bmod p$ is a ring homomorphism means that for each monomial Y_j, we have

$$(Y_j(\sigma_1, \ldots, \sigma_k))(a) \bmod p = Y_j(\sigma_1(a), \ldots, \sigma_k(a)) \bmod p.$$

By correctness of individual signatures, we have $\sigma_i(a) \bmod p = m_i$ for $i \in \{1, \ldots, k\}$. It follows that

$$\begin{aligned}\sigma'(a) \bmod p &= \sum_j c_j (Y_j(\sigma_1, \ldots, \sigma_k))(a) \bmod p \\ &= \sum_j c_j Y_j(\sigma_1(a), \ldots, \sigma_k(a)) \bmod p = \sum_j c_j Y_j(m_1, \ldots, m_k) = f(\vec{m}).\end{aligned}$$

(c) By correctness of individual signatures, we have $\sigma_i(b) \bmod q = \alpha_i$ for $i \in \{1, \ldots, k\}$. The same analysis as in part (b) above, using the ring homomorphism $h(x) \mapsto h(b) \bmod q$, shows that

$$\sigma'(b) \bmod q = \sum_j c_j Y_j(\alpha_1, \ldots, \alpha_k) \bmod q = \omega_\tau(\langle f \rangle). \qquad \square$$

6.1 Length efficiency

With overwhelming probability, signatures output by $\mathsf{Sign}(\mathsf{sk}, \cdot, \cdot, \cdot)$ have norm at most $\nu\sqrt{n}$ and therefore bit length at most $n \lg(\nu\sqrt{n})$. As shown in the proof of Lemma 6.1 above, applying Evaluate to a polynomial function $f \colon \mathbb{F}_p^k \to \mathbb{F}_p$ and k signatures generated by Sign outputs a signature σ whose norm is at most $\ell \cdot y \cdot \gamma_F^{d-1} \cdot (\nu\sqrt{n})^d$ with overwhelming probability. Since $\ell \leq k^d$, the bit length of this derived signature σ satisfies

$$\mathrm{len}(\sigma) \leq n \cdot d \lg k + n \lg\left(y \cdot \gamma_F^{d-1} \cdot \nu\sqrt{n}\right).$$

Since we require $\gamma_F = \mathrm{poly}(n)$, $y = \mathrm{poly}(n)$, and $d = O(1)$, it follows that the bit length of the derived signature depends logarithmically on k, and thus we conclude that the scheme is length efficient.

6.2 Unforgeability

As in our linearly homomorphic scheme from Section 4, an adversary that can forge a signature in the above system can be used to find a short vector in the lattice used to authenticate functions, which in this case is the ideal $\mathfrak{q}$.

Theorem 6.2. *For fixed n, let F_n be the polynomial chosen in Step (1) of the* Setup *algorithm above, and let $\mathcal{L}_n$ be the distribution of ideals $\mathfrak{q}$ output by the Smart-Vercauteren algorithm when given input polynomial $F_n(x)$ and parameter $\delta = n$. Let $\mathcal{L}_F$ be the ensemble $\{\mathcal{L}_n\}$. If $\mathcal{L}_F$-$\mathsf{SIS}_{n,\beta}$ is infeasible for*

$$\beta = 2 \cdot \tbinom{k+d}{d} \cdot y \cdot \gamma_{F_n}^{3d-1} \left(n^3 \log n\right)^d,$$

then the polynomially homomorphic signature scheme defined above is unforgeable in the random oracle model.

The proof of this theorem will follow from the security of our abstract system that generalizes the polynomially homomorphic scheme; see Section 7.4 for details. While Theorem 6.2 gives a concrete security result for our system, the distribution $\mathcal{L}_F$ of prime ideals output by the Smart-Vercauteren algorithm is not well understood. It is an open problem to modify the system to use ideals sampled from a distribution that admits a random self-reduction.

6.3 Privacy

The homomorphic signature scheme described above is not weakly context hiding in the sense of Definition 2.3 whenever the degree d of admissible polynomials is greater than 1. To see why, consider an attacker that outputs two data sets

$$\vec{m}_0^* := (0,0) \qquad \text{and} \qquad \vec{m}_1^* := (0,1),$$

each containing two messages in $(R/\mathfrak{p}) \cong \mathbb{F}_p$. The attacker also outputs the function $f(x,y) := x \cdot y$, which is a valid function to request since $f(\vec{m}_0^*) = f(\vec{m}_1^*)$.

The challenger chooses a random bit b in $\{0,1\}$ and generates signatures σ_1, σ_2 in R for the two messages in $\vec{m}_b^*$. It gives the attacker $\sigma := \sigma_1 \cdot \sigma_2$.

Now, when $b = 0$ both σ_1 and σ_2 are in $\mathfrak{p}$ and therefore the derived signature $\sigma = \sigma_1\sigma_2$ is in $\mathfrak{p}^2$. However, when $b = 1$ we know that $\sigma_2 \notin \mathfrak{p}$ and therefore $\sigma \in \mathfrak{p}^2$ only if $\sigma_1 \in \mathfrak{p}^2$. But σ_1 is in $\mathfrak{p}^2$ with probability at most $1/2$. In other words, $\Pr[\sigma \in \mathfrak{p}^2] = 1$ when $b = 0$, but $\Pr[\sigma \in \mathfrak{p}^2] \leq 1/2$ when $b = 1$. Therefore, an adversary that outputs $b' = 0$ if $\sigma \in \mathfrak{p}^2$ and $b' = 1$ otherwise has advantage at least $1/2$ in distinguishing $\vec{m}_0^*$ from $\vec{m}_1^*$ just given σ. Consequently the scheme is not weakly context hiding.

6.4 Using small fields

The signature scheme described above signs messages defined over a finite field $\mathbb{F}_p$, where p is exponential in n. In some cases we may wish to authenticate polynomial functions of data defined over a smaller field, as we can for linear computations using the scheme of Section 4.

To do this, instead of generating the ideal $\mathfrak{p}$ using the algorithm of Smart and Vercauteren, we simply select *any* prime ideal $\mathfrak{p}$ of characteristic $p < n^{1.5}$. For example, for our "preferred" choice of field defined by $F(x) = x^n + 1$ with n a power of 2, the ideal $\mathfrak{p} = (1+x)\mathcal{O}_K$ is a degree one prime of characteristic 2. This choice of $F(x)$ and $\mathfrak{p}$ can therefore be used to authenticate polynomial computations on bits, i.e., AND and XOR.

We now consider how our new choice of $\mathfrak{p}$ affects the system. Note that we place no restriction on the degree e of $\mathfrak{p}$, so in particular the residue field $R/\mathfrak{p} = \mathbb{F}_{p^e}$ that defines the message space need not be of prime cardinality. (We generate the prime $\mathfrak{q}$ as before.)

- Representation of $\mathfrak{p}$: We require that p be relatively prime to the conductor of $R = \mathbb{Z}[x]/(F(x))$ in $\mathcal{O}_K$. By Kummer-Dedekind, we can represent $\mathfrak{p}$ as $(p, g(x))$, where $g(x) \bmod p$ is one of the irreducible factors of $F(x) \bmod p$.
- Short basis of $\mathfrak{p} \cdot \mathfrak{q}$: We can apply the algorithm of Lemma 3.1 to the set $\{pg_{\mathfrak{q}}, pg_{\mathfrak{q}}x, \ldots, pg_{\mathfrak{q}}x^{n-1}\}$ to produce a basis $\mathbf{T}$ of $\mathfrak{p} \cdot \mathfrak{q}$. Since $\|g_{\mathfrak{q}}\| < n^{1.5}$ and $p < n^{1.5}$, we have $\|\widetilde{\mathbf{T}}\| \leq \gamma_F \cdot n^3$.
- Message space: the message space is the field $\mathbb{F}_{p^e}$, where e is the degree of $\mathfrak{p}$. (Of course it is possible to restrict to the subfield $\mathbb{F}_p$.)
- Admissible functions: If we set $y = p/2$, then the set of admissible functions $\mathcal{F}$ consists of *all* polynomials in $\mathbb{F}_p[x_1, \ldots, x_k]$.
- Signing and verifying: In Step (2) of Sign we choose $h(x)$ so that $h(x) \bmod \mathfrak{p} = m$, and in Step (1b) of Verify we check that $\sigma(x) \bmod \mathfrak{p} = m$.

The proofs of Lemma 6.1 and Theorem 6.2 can easily be adapted to show that the statements hold for this modified system.

7 An Abstract Construction

In this section we describe an abstract homomorphic signature scheme that captures the key properties of both the linearly homomorphic scheme of Section 4 and the polynomially homomorphic scheme of Section 6. We prove that the abstract scheme is correct and unforgeable, and we prove unforgeability of our two concrete schemes by showing that they are special cases of the abstract scheme.

The abstract signature scheme uses two n-dimensional integer lattices Λ_1 and Λ_2. The message space is $\mathbb{Z}^n/\Lambda_1$ and signatures are short vectors $\sigma \in \mathbb{Z}^n$ that live in a particular coset of $\Lambda_1 \cap \Lambda_2$. As before, we let $\mathcal{F}$ denote the set of admissible functions on messages.

In our concrete schemes we constructed the Evaluate algorithm by "lifting" admissible functions on messages (defined over $\mathbb{F}_p$) to functions on signatures (defined over $\mathbb{Z}$). To formalize this lifting for our abstract scheme, we first fix linear quotient maps

$$\phi_1 \colon \mathbb{Z}^n \to \mathbb{Z}^n/\Lambda_1 \cong \mathbb{F}_p^r \quad \text{and} \quad \phi_2 \colon \mathbb{Z}^n \to \mathbb{Z}^n/\Lambda_2 \cong \mathbb{F}_q^s$$

(for some r, s). To "lift" a function $f \in \mathcal{F}$, we fix a set $\mathcal{F}_0 = \{f_j\}$ whose $\mathbb{F}_p$-linear span contains all functions in $\mathcal{F}$. We can then write $f = \sum c_j f_j$ for unique $c_j \in \mathbb{F}_p$, interpret the c_i as integers in $(-p/2, p/2]$, and use the vector $(c_1, \ldots, c_\ell) \in \mathbb{Z}^\ell$ as the encoding $\langle f \rangle$ of f. We also choose a set of functions $\mathcal{H}_0 = \{h_j \colon (\mathbb{Z}^n)^k \to \mathbb{Z}^n\}$ that operate on signatures and are "compatible" with the f_j via ϕ_1, in the sense that

$$\phi_1\big(h_j(x_1, \ldots, x_k)\big) = f_j\big(\phi_1(x_1), \ldots, \phi_1(x_k)\big) \quad \text{for all } \vec{x} \in (\mathbb{Z}^n)^k.$$

Since we interpret the coefficients c_j in $\mathbb{F}_p$ as integers in $(-p/2, p/2]$, there is a well-defined lifting of functions $f \in \mathcal{F}$ to functions on signatures, given by $\sum c_j f_j \mapsto \sum c_j h_j$.

In addition, in the concrete schemes we verified signatures using the hash function ω_τ, which at its core "translates" a function $f \in \mathcal{F}$ to a function on the $\alpha_i \in \mathbb{Z}^n/\Lambda_2$ used to sign messages. In our abstract scheme we achieve this functionality by defining a third set of functions $\mathcal{G}_0 = \{g_j\}$ on $\mathbb{Z}^n/\Lambda_2$ that are "compatible" with the h_j via ϕ_2 in the same sense as described above. We can now map functions $f \in \mathcal{F}$ to functions on the α_i by mapping $\sum c_j f_j \mapsto \sum c_j g_j$, and use the resulting function in the verification algorithm.

For the Evaluate function to be correct we must obtain a short vector as output. In particular, we must bound the amount that the functions h_i "expand" their inputs. The following definition gives us a framework for measuring this "expansion."

Definition 7.1. Let $\Lambda, \Lambda' \subset \mathbb{R}^n$ be lattices. We say that a function $f : \Lambda^k \to \Lambda'$ is *homogeneous of degree* d if $f(\lambda\vec{x}) = \lambda^d f(\vec{x})$ for all $\lambda \in \mathbb{Z}$ and $\vec{x} \in \Lambda^k$.

For a homogeneous function of degree d, we define the *expansion factor* of f to be

$$|f| := \sup_{\vec{x}\in\Lambda^k} \frac{\|f(\vec{x})\|}{\max_i \|x_i\|^d}. \tag{7.1}$$

For any given f, it is possible that $|f|$ is infinite; however, in our construction we will use only functions f with finite expansion factor. In particular, for the *projection functions* π_i defined by $\pi_i(x_1, \ldots, x_k) = x_i$, we have $|\pi_i| = 1$.

7.1 The abstract scheme

We now describe our abstract scheme. It encompasses both our linearly homomorphic system over random q-ary lattices and our polynomially homomorphic system over ideal lattices.

$\mathsf{Setup}(1^n, k)$. On input a security parameter n and a maximum data set size k, do the following:

1. Choose two primes p, q (not necessarily distinct). Generate two lattices $\Lambda_1, \Lambda_2 \subset \mathbb{Z}^n$, such that $p\mathbb{Z}^n \subset \Lambda_1$ and $q\mathbb{Z}^n \subset \Lambda_2$, along with a short basis $\mathbf{T}$ of $\Lambda_1 \cap \Lambda_2$. Choose $\nu \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$.
2. Let $r = \dim_{\mathbb{F}_p}(\mathbb{Z}^n/\Lambda_1)$, and define a linear map $\phi_1 : \mathbb{Z}^n \to \mathbb{F}_p^r$ with kernel Λ_1.
   Let $s = \dim_{\mathbb{F}_q}(\mathbb{Z}^n/\Lambda_2)$, and define a linear map $\phi_2 : \mathbb{Z}^n \to \mathbb{F}_q^s$ with kernel Λ_2.
   The maps ϕ_1 and ϕ_2 project from $\mathbb{Z}^n$ onto $\mathbb{Z}^n/\Lambda_1$ and $\mathbb{Z}^n/\Lambda_2$ respectively.
   We require that for any $x \in \mathbb{F}_p^r$, we can efficiently compute an (arbitrary) element of $\phi_1^{-1}(x)$ in $\mathbb{Z}^n$, and similarly for any $y \in \mathbb{F}_q^s$ with respect to ϕ_2. (This property holds trivially for all of our instantiations.)
3. Choose an integer $\ell \geq k$ and define three sets of functions of cardinality ℓ:
   - $\mathcal{F}_0 = \{f_1, \ldots, f_\ell\}$ is a linearly independent set of functions from $(\mathbb{F}_p^r)^k$ to $\mathbb{F}_p^r$. These functions operate on k-tuples of messages in $\mathbb{F}_p^r$ and linearly span the set of admissible functions $\mathcal{F}$.
   - $\mathcal{G}_0 = \{g_1, \ldots, g_\ell\}$ is a set of functions from $(\mathbb{F}_q^s)^k$ to $\mathbb{F}_q^s$. These functions will be used to define the hash ω_τ.
   - $\mathcal{H}_0 = \{h_1, \ldots, h_\ell\}$ is a set of homogeneous functions from $(\mathbb{Z}^n)^k$ to $\mathbb{Z}^n$. These functions operate on signatures. We let $d := \max\{\deg h_j\}$, and we require that $|h_j| < \infty$ for all j.

   We require that the functions f_j, g_j, h_j be compatible with the quotient maps ϕ_1, ϕ_2. Specifically, for $j = 1, \ldots, \ell$, we require that for all $\vec{x} \in (\mathbb{Z}^n)^k$, we have

   $$f_j \circ (\phi_1)^k = \phi_1 \circ h_j \qquad \text{and} \qquad g_j \circ (\phi_2)^k = \phi_2 \circ h_j. \tag{7.2}$$

   We also require that for $i = 1, \ldots, k$, the functions f_i, g_i, h_i are the projection functions π_i on $(\mathbb{F}_p^r)^k$, $(\mathbb{F}_q^s)^k$, and $(\mathbb{Z}^n)^k$, respectively.[5]
4. Define a set $\mathcal{C} \subset \mathbb{Z}$ with $\{0, 1\} \subset \mathcal{C}$ such that the map $c \mapsto (c \bmod p)$ is injective on $\mathcal{C}$.
5. Let $H\colon \{0,1\}^* \to (\mathbb{F}_q^s)^k$ be a hash function (modeled as a random oracle).
6. Output the public key $\mathsf{pk} = (\Lambda_1, \Lambda_2, \phi_1, \phi_2, \mathcal{C}, \mathcal{F}_0, \mathcal{G}_0, \mathcal{H}_0, H, \nu, d)$ and the secret key $\mathsf{sk} = \mathbf{T}$.

[5] By abuse of notation, we call all of these projection functions π_i.

The public key pk defines the following system parameters:

- The message space $\mathcal{M}$ is $\mathbb{F}_p^r$ and signatures are short vectors in $\mathbb{Z}^n$.
- The set of admissible functions is $\mathcal{F} := \left\{ \sum_{j=1}^{\ell} c_j f_j : c_j \in \mathcal{C},\ f_j \in \mathcal{F}_0 \right\}$.
- The encoding of a function $f = \sum c_j f_j$ is the vector $\langle f \rangle := (c_1, \ldots, c_\ell) \in \mathcal{C}^\ell$.
  Since the mod p map is injective on $\mathcal{C}$ and the set $\mathcal{F}_0$ is linearly independent, every function in $\mathcal{F}$ has a unique encoding.
- To evaluate the hash function ω_τ on an encoding $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathcal{C}^\ell$, do the following:
  (a) For $i = 1, \ldots, k$, compute $\alpha_i \leftarrow H(\tau \| i)$.
  (b) Define a function g on $(\mathbb{F}_q^s)^k$ by $g := \sum_{j=1}^{\ell} c_j g_j$.
  (c) Define $\omega_\tau(\langle f \rangle) := g(\alpha_1, \ldots, \alpha_k) \quad \in \mathbb{F}_q^s$.
- The public key also defines a parameter $\lambda := \ell \cdot \max\{|c| : c \in \mathcal{C}\} \cdot \max\{|h_j| : h_j \in \mathcal{H}_0\}$.
  Since $1 \in \mathcal{C}$ and $\pi_i \in \mathcal{H}_0$, we have $\lambda \geq 1$.

$\mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. On input a secret key sk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathcal{M}$, and an index i, do:

1. Compute $\alpha_i \leftarrow H(\tau \| i)$.
2. Compute $\mathbf{t} \in \mathbb{Z}^n$ such that $\phi_1(\mathbf{t}) = m$ and $\phi_2(\mathbf{t}) = \alpha_i$.
3. Output $\sigma \leftarrow \mathsf{SamplePre}(\Lambda_1 \cap \Lambda_2, \mathbf{T}, \mathbf{t}, \nu) \quad \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$.

$\mathsf{Verify}(\mathsf{pk}, \tau, m, \sigma, f)$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_p^r$, a signature $\sigma \in \mathbb{Z}^n$, and a function $f \in \mathcal{F}$, do:

1. If all of the following conditions hold, output 1 (accept); otherwise output 0 (reject):
   (a) $\|\sigma\| \leq \lambda \cdot (\nu\sqrt{n})^d$.
   (b) $\phi_1(\sigma) = m \quad \in \mathbb{Z}^n/\Lambda_1 = \mathbb{F}_p^r$.
   (c) $\phi_2(\sigma) = \omega_\tau(\langle f \rangle) \quad \in \mathbb{Z}^n/\Lambda_2 = \mathbb{F}_q^s$.

$\mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma})$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a function $f \in \mathcal{F}$ encoded as $\langle f \rangle = (c_1, \ldots, c_\ell) \in \mathcal{C}^\ell$, and a tuple of signatures $\vec{\sigma} \in (\mathbb{Z}^n)^k$, do:

1. Define a function h on $(\mathbb{Z}^n)^k$ by $h := \sum c_j h_j$.
2. Output $h(\vec{\sigma})$.

Lemma 7.2. *The homomorphic signature scheme defined above is correct with overwhelming probability.*

Proof. Let $\tau \in \{0,1\}^n$, $m \in \mathcal{M}$, and $i \in \{1, \ldots, k\}$, and set $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. We check the three verification conditions.

(a) By Lemma 3.6, we have $\|\sigma\| \leq \nu\sqrt{n}$ with overwhelming probability.

(b) By correctness of SamplePre, we have $\sigma \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$. By definition of $\mathbf{t}$, we have $\phi_1(\mathbf{t}) = m$. Since ϕ_1 is linear and $\Lambda_1 \subset \ker \phi_1$, we see that $\phi_1(\sigma) = m$.

(c) By our requirement in Step (3) of Setup and our definition of ω_τ, we have $\omega_\tau(\langle \pi_i \rangle) = \alpha_i$. As above, we have $\sigma \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$. By definition of $\mathbf{t}$, we have $\phi(\mathbf{t}) = \alpha_i$. Since ϕ_2 is linear and $\Lambda_2 \subset \ker \phi_2$, we see that $\phi_2(\sigma) = \alpha_i$.

Now let $\tau \in \{0,1\}^n$, $\vec{m} = (m_1, \ldots, m_k) \in \mathcal{M}^k$, and $f \in \mathcal{F}$ encoded as $(c_1, \ldots, c_\ell) \in \mathcal{C}^\ell$. Let $\vec{\sigma} = (\sigma_1, \ldots, \sigma_k)$ with $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, \tau, m_i, i)$. Then by our definition of Evaluate, we have

$$\sigma' := \mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma}) = \textstyle\sum_{j=1}^{\ell} c_j h_j(\vec{\sigma}).$$

We check the three verification conditions for the signature σ' on the message $f(\vec{m})$.

(a) By the definition of expansion factor (7.1), we have $\|h_j(\vec{\sigma})\| \le |h_j|^{\deg h_j} \max_i(\|\sigma_i\|^{\deg h_j})$. Multiplying by c_j and adding up ℓ terms, we obtain

$$\|\sigma'\| \;\le\; \ell \cdot \max\{c : c \in \mathcal{C}\} \cdot \max\{|h_j| : h_j \in \mathcal{H}_0\} \cdot \max\{\|\sigma_i\|^d : \sigma_i \in \vec{\sigma}\}$$

By Lemma 3.6 and the definition of λ, the right hand side is less than $\lambda \cdot (\nu\sqrt{n})^d$ with overwhelming probability.

(b) By correctness of individual signatures, we have $\phi_1(\sigma_i) = m_i$ for all $i \in \{1, \ldots, k\}$. It follows that

$$\begin{aligned}\phi_1(\sigma') \;=\; \phi_1\left(\sum_j c_j h_j(\vec{\sigma})\right) \;&=\; \sum_j c_j \cdot \phi_1 \circ h_j(\vec{\sigma}) \;=\; \sum_j c_j \cdot f_j \circ \phi_1^k(\vec{\sigma}) \\ &= \sum_j c_j \cdot f_j(\phi_1(\sigma_1), \ldots, \phi_1(\sigma_k)) \;=\; \sum_j c_j f_j(m_1, \ldots, m_k) \;=\; f(\vec{m}).\end{aligned}$$

Note that the last equality on the first line follows from the compatibility condition (7.2).

(c) By correctness of individual signatures, we have $\phi_2(\sigma_i) = \alpha_i$ for all $i \in \{1, \ldots, k\}$. An analysis identical to that in part (b) above, also using (7.2), shows that $\phi_2(\sigma') = g(\alpha_1, \ldots, \alpha_k) = \omega_\tau(\langle f \rangle)$. □

7.2 Unforgeability of the abstract scheme

Unforgeability of our abstract scheme (in the random oracle model) is based on the difficulty of finding a short nonzero vector in Λ_2; specifically, a vector of length at most $2\lambda \cdot (\nu\sqrt{n})^d$.

Our proof of security requires that functions in $\mathcal{F}$ do not map too many inputs to one output. In particular, $\mathcal{F}$ cannot contain any nonzero constant functions. This property, which we now define formally, is easily shown to hold for the families of functions that we consider.

Definition 7.3. Let V be a finite-dimensional vector space over a finite field $\mathbb{F}_q$, and let $f : V^k \to V$ be a (not necessarily linear) function. We say f is *d-solution-bounded* for an integer d if for every $x \in V$, the number of solutions in V^k to $f(\vec{y}) = x$ is at most $d \cdot |V|^{k-1}$. We say that a set $\mathcal{F}$ of functions is d-solution-bounded if every nonzero $f \in \mathcal{F}$ is d-solution-bounded.

We can now prove our security theorem. Our reduction is tight; in particular, we can use the same challenge lattice to answer all queries, and thus we do not need to guess which data set the adversary will forge a signature on.

Theorem 7.4. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be the distribution ensembles of lattices sampled in Step 1 of the* Setup *algorithm. Let ν be a parameter such that for $\Lambda_1 \leftarrow \mathcal{L}_1$ and $\Lambda_2 \leftarrow \mathcal{L}_2$, with overwhelming probability we have $\nu > \eta_\epsilon(\Lambda_1 \cap \Lambda_2)$ for negligible ϵ, and assume that this ν is the value chosen in Step 1 of the* Setup *algorithm for all Λ_1, Λ_2. Suppose furthermore that for $\Lambda_1 \leftarrow \mathcal{L}_1$ and $\Lambda_2 \leftarrow \mathcal{L}_2$, we have $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$ with overwhelming probability. Finally, suppose that $\mathcal{F}$ is $\lfloor q/2 \rfloor$-solution-bounded.*

Suppose that there is an efficient algorithm $\mathcal{G}$ that samples a lattice Λ_1 from a distribution ensemble statistically close to $\mathcal{L}_1$, along with a basis $\mathbf{T}_1$ of Λ_1 with $\|\widetilde{\mathbf{T}_1}\| \le \nu/\omega(\sqrt{\log n})$. If the $\mathcal{L}_2$-$\mathsf{SIS}_{n,2\lambda\cdot(\nu\sqrt{n})^d}$ problem is infeasible, then the abstract homomorphic signature scheme described above is unforgeable in the random oracle model.

Proof. Let $\mathcal{A}$ be a polynomial-time adversary that plays the security game of Definition 2.1. Given a challenge lattice $\Lambda_2 \leftarrow \mathcal{L}_2$, we construct an algorithm $\mathcal{B}$ that solves the $\mathcal{L}_2$-$\mathsf{SIS}_{n,2\lambda\cdot(\nu\sqrt{n})^d}$ problem. Algorithm $\mathcal{B}$ simulates the Setup and Sign algorithms and the hash function H as follows:

Setup: Use the algorithm $\mathcal{G}$ to generate a lattice Λ_1 along with a short basis $\mathbf{T}_1$ of Λ_1. Choose all other parameters as in the real Setup algorithm, using the challenge lattice for Λ_2.

Hash function H: On input $\tau\|i$, if $\tau\|i$ has already been queried to H, then return $H(\tau\|i)$. Otherwise choose $\sigma_i \leftarrow \mathcal{D}_{\mathbb{Z}^n,\nu}$ and set $H(\tau\|i) := \phi_2(\sigma_i)$.

Signing queries: When $\mathcal{A}$ queries the data set $(m_1,\ldots,m_k) \in (\mathbb{F}_p^r)^k$, do the following:

1. Choose a random $\tau \xleftarrow{\mathrm{R}} \{0,1\}^n$. If $\tau\|i$ has already been queried to the hash function H for some i in $1,\ldots,k$, then abort. (The simulation has failed.)
2. For $i = 1,\ldots,k$, choose $\mathbf{t}_i \in \phi_1^{-1}(m_i)$, and compute $\sigma_i \leftarrow \mathsf{SamplePre}(\Lambda_1, \mathbf{T}_1, \mathbf{t}_i, \nu)$.
3. Define $H(\tau\|i) := \phi_2(\sigma_i)$ for $i = 1,\ldots,k$.
4. Give to $\mathcal{A}$ the tag τ and the signatures $\sigma_1,\ldots,\sigma_k$.

Eventually $\mathcal{A}$ outputs a tag τ^*, a message m^*, a function f encoded as $\langle f\rangle = (c_1,\ldots,c_\ell) \in \mathcal{C}^\ell$, and a signature σ^*. We may assume without loss of generality that $\tau^*\|i$ have already been queried to the hash function for $i = 1,\ldots,k$; let σ_i be the value chosen when programming $H(\tau^*\|i)$, and let $\vec{\sigma} = (\sigma_1,\ldots,\sigma_k)$. Let $\sigma_f := \sum_i c_i h_i(\vec{\sigma})$. Algorithm $\mathcal{B}$ outputs $\sigma^* - \sigma_f$.

We first show that the output of the simulator is distributed (up to negligible statistical distance) as in the real signature scheme. To begin, since the simulator chooses random tags from $\{0,1\}^n$ when signing and $\mathcal{A}$ runs in polynomial time, the probability that the simulator aborts is negligible in n, so we may assume that the simulator does not abort. Next, since $\nu > \eta_\epsilon(\Lambda_1 \cap \Lambda_2)$ for negligible ϵ, by Lemma 3.9 the vectors σ_i chosen in both hashing and signing queries are statistically close to uniform modulo Λ_2, and thus the output of the hash function is indistinguishable from random. By Theorem 3.3, the σ_i in the real scheme are distributed as $\mathcal{D}_{\Lambda_1\cap\Lambda_2+\mathbf{t},\nu}$ for $\mathbf{t} \in \phi_1^{-1}(m_i) \cap \phi_2^{-1}(\alpha_i)$. The simulated σ_i, on the other hand, are distributed as $\mathcal{D}_{\Lambda_1+\mathbf{t},\nu}$ for $\mathbf{t} \in \phi_1^{-1}(m_i)$, conditioned on $\phi_2(\sigma_i) = \alpha_i$. By a straightforward generalization of [20, Lemma 5.2], these two distributions are identical.

We now show that if $\mathcal{A}$ outputs a valid forgery, then with probability at least $1/2$, the vector $\sigma^* - \sigma_f$ output by $\mathcal{B}$ is a nonzero vector in Λ_2 of length at most $2\lambda\cdot(\nu\sqrt{n})^d$.

Suppose $\mathcal{A}$ outputs a type 2 forgery, so the simulator has generated signatures $\vec{\sigma} = (\sigma_1,\ldots,\sigma_k)$ on messages $\vec{m} = (m_1,\ldots,m_k)$ using the tag τ^*. First observe that the verification condition (1a) implies that $\|\sigma^*\|$ and $\|\sigma_f\|$ are both less than $\lambda\cdot(\nu\sqrt{n})^d$, and therefore $\|\sigma^* - \sigma_f\| \le 2\lambda\cdot(\nu\sqrt{n})^d$. Next observe that if the forgery is valid, then $m^* \neq f(\vec{m})$. The verification condition (1b) implies that $\phi_1(\sigma^*) - \phi_1(\sigma_f) = m^* - f(\vec{m}) \neq 0$, and thus linearity of ϕ_1 implies that $\sigma^* - \sigma_f \neq 0$. On the other hand, verification condition (1c) implies that $\phi_2(\sigma^*) = \phi_2(\sigma_f) = \sum c_i g_i(\vec{\alpha})$, and thus $\sigma^* - \sigma_f \in \Lambda_2$.

Now suppose $\mathcal{A}$ outputs a type 1 forgery, so the adversary has not obtained signatures on messages with tag τ^*. Without loss of generality, we can assume that $\tau^*\|i$ were queried to the hash function at some point. Let $\vec{\sigma} = (\sigma_1,\ldots,\sigma_k)$ be the vectors chosen by the simulator when computing $H(\tau^*\|i)$. Define $m_i = \phi_1(\sigma_i)$; by Lemma 3.9 the m_i are uniformly random in $(\mathbb{F}_p^r)^k$. If $f(\vec{m}) \neq m^*$, then by the same reasoning as above the vector $\sigma^* - \sigma_f$ is a nonzero vector in Λ_2 of the required length. Finally, we observe that since f is $\lfloor q/2\rfloor$-solution-bounded, the number of solutions in $(\mathbb{F}_q^s)^k$ to $f(\vec{m}) = m^*$ is at most $q^{sk-s+1}/2$. Since the total number of elements $\vec{m} \in (\mathbb{F}_q^s)^k$ is q^{sk}, the probability that a random $\vec{m}$ is a solution to $f(\vec{m}) = m^*$ is at most $1/2$. □

7.3 Proof of unforgeability for linear scheme

In this section we prove that the linearly homomorphic signature scheme of Section 4 is unforgeable. We first translate the concrete description of the scheme given in Section 4 into the abstract framework of Section 7.1. We then show that the scheme satisfies the hypotheses of Theorem 7.4, and unforgeability follows.

We begin by showing how the Setup algorithm in Section 4 instantiates the abstract Setup algorithm of Section 7.1. The numbered items below correspond to the steps of the abstract Setup algorithm.

1. We choose two primes $p, q \in \mathrm{poly}(n)$ with $p \leq \sqrt{q}/n$, and define $\ell := \lfloor n/6 \log q \rfloor$. The lattice Λ_1 is $p\mathbb{Z}^n$ and Λ_2 is $\Lambda_q^\perp(\mathbf{A})$ for a matrix $\mathbf{A} \in \mathbb{F}_q^{\ell \times n}$ generated by TrapGen. We set $\nu := p \cdot \sqrt{n \log q} \cdot \log n$.

2. Let $r = n$, and define the quotient map $\phi_1 : \mathbb{Z}^n \to \mathbb{F}_p^n$ by $\mathbf{v} \mapsto (\mathbf{v} \bmod p)$.
   Let $s = \ell$, and define the quotient map $\phi_2 : \mathbb{Z}^n \to \mathbb{F}_q^\ell$ by $\mathbf{v} \mapsto (\mathbf{A} \cdot \mathbf{v} \bmod q)$.

3. For an integer $k = \ell \leq \sqrt{q}/np$, the three sets $\mathcal{F}_0, \mathcal{G}_0, \mathcal{H}_0$ are the sets of projection maps π_i on $(\mathbb{F}_p^n)^k$, $(\mathbb{F}_q^\ell)^k$, and $(\mathbb{Z}^n)^k$, respectively. We have $d = 1$ and $|h_i| = 1$ for all i.

4. The set $\mathcal{C}$ is integers in $(-p/2, p/2]$.

The parameter λ is equal to $k \cdot p/2$.

Proof of Theorem 4.2. By Theorem 3.2, the distribution of matrices $\mathbf{A}$ generated by TrapGen is statistically close to uniform over $\mathbb{F}_q^{\ell \times n}$. Thus the distribution of challenges for the $\mathcal{L}_2$-$\mathsf{SIS}_{n,2\lambda\cdot\nu\sqrt{n}}$ problem is statistically close to the distribution of challenges for the $\mathsf{SIS}_{q,n,k\cdot p\cdot\nu\sqrt{n}}$ problem.

We now verify the hypotheses of Theorem 7.4. First, since p and q are relatively prime, we have $\Lambda_q^\perp(\mathbf{A}) + p\mathbb{Z}^n = \mathbb{Z}^n$ and $\Lambda_q^\perp(\mathbf{A}) \cap p\mathbb{Z}^n = p\Lambda_q^\perp(\mathbf{A})$. By Lemma 3.5, with overwhelming probability we have $\eta_\epsilon(\Lambda_2) < \log n$ for negligible ϵ. Since $\eta_\epsilon(\Lambda_1 \cap \Lambda_2) = p \cdot \eta_\epsilon(\Lambda_2)$, with overwhelming probability we have $\nu > \eta_\epsilon(\Lambda_1 \cap \Lambda_2)$ for negligible ϵ. In addition, the standard basis of $p\mathbb{Z}^n$ has Gram-Schmidt norm p, which is less than $\nu/\omega(\sqrt{\log n})$. Finally, it is easy to see that for any $\mathbb{F}_q$-vector space V, any nonzero $\mathbb{F}_q$-linear function $f : V^k \to V$ is 1-solution-bounded. □

7.4 Proof of unforgeability for polynomial scheme

In this section we prove that the polynomially homomorphic signature scheme of Section 6 is unforgeable. We first translate the concrete description of the scheme given in Section 6 into the abstract framework of Section 7.1. We then show that the scheme satisfies the hypotheses of Theorem 7.4, and unforgeability follows.

We begin by showing how the Setup algorithm in Section 6 instantiates the abstract Setup algorithm of Section 7.1. The numbered items below correspond to the steps of the abstract Setup algorithm.

1. The lattices Λ_1 and Λ_2 are ideals $\mathfrak{p} = (p, x-a)$ and $\mathfrak{q} = (q, x-b)$ in the number ring $R = \mathbb{Z}[x]/(F(x))$; we identify R with $\mathbb{Z}^n$ via the coefficient embedding. We generate $\mathfrak{p}$ and $\mathfrak{q}$ using $\mathsf{PrincGen}(F, n)$, which also outputs generators $g_\mathfrak{p}, g_\mathfrak{q}$ of $\mathfrak{p}, \mathfrak{q}$, respectively. We obtain a basis $\mathbf{T}$ of $\mathfrak{p} \cdot \mathfrak{q}$ by applying the algorithm of Lemma 3.1 to the set $\{g_\mathfrak{p} g_\mathfrak{q}, g_\mathfrak{p} g_\mathfrak{q} x, \ldots, g_\mathfrak{p} g_\mathfrak{q} x^{n-1}\}$. We set $\nu = \gamma_F^2 \cdot n^3 \log n$.

2. Let $r = s = 1$. Define the quotient map $\phi_1 : R \to \mathbb{F}_p$ by $z(x) \mapsto z(a) \bmod p$.
   Define the quotient map $\phi_2 : R \to \mathbb{F}_q$ by $z(x) \mapsto z(b) \bmod q$.

3. Choose an integer $d = O(1)$ and let $\ell := \binom{k+d}{d} - 1$. We define the sets $\mathcal{F}_0, \mathcal{G}_0, \mathcal{H}_0$ to be sets of all ℓ monic, non-constant k-variable monomial functions of degree at most d on $\mathbb{F}_p$, $\mathbb{F}_q$, and R, respectively, using the lexicographic ordering.

4. Define $\mathcal{C} := \{-y, \ldots, y\}$ for $y = \text{poly}(n)$.

To compute the parameter λ, we observe that for each monomial function h_i on R, we have $|h_i| \leq \gamma_F^{\deg h_i - 1}$. Thus we have $\lambda \leq \ell \cdot y \cdot \gamma_F^{d-1}$. Since we assumed that $\gamma_F = \text{poly}(n)$, we have $\nu = \text{poly}(n)$.

Proof of Theorem 6.2. We verify the hypotheses of Theorem 7.4.

Since $\mathfrak{p}$ and $\mathfrak{q}$ are distinct prime ideals, we have $\mathfrak{p} \cap \mathfrak{q} = \mathfrak{p} \cdot \mathfrak{q}$ and $\mathfrak{p} + \mathfrak{q} = R$. By Theorem 5.1 the generators $g_\mathfrak{p}, g_\mathfrak{q}$ have norm at most $n^{1.5}$, and therefore $\|g_\mathfrak{p} g_\mathfrak{q} x^i\| \leq \gamma_F^2 \cdot n^3$ for $i = 0, \ldots, n-1$. It follows from Lemma 3.1 that the basis $\mathbf{T}$ has $\|\widetilde{\mathbf{T}}\| \leq \gamma_F^2 n^3$. Since $\nu \geq \|\widetilde{\mathbf{T}}\| \cdot \log n$, by Lemma 3.4 with overwhelming probability we have $\nu > \eta_\epsilon(\mathfrak{p} \cdot \mathfrak{q})$ for negligible ϵ.

Next, note that nonzero functions in $\mathcal{F}$ are non-constant polynomial functions in k variables over $\mathbb{F}_p$ of degree at most d. By Schwartz's Lemma [38], for any such polynomial f and any $y \in \mathbb{F}_p$ there are at most $d \cdot p^{k-1}$ solutions $\vec{x} \in \mathbb{F}_p^k$ to $f(\vec{x}) - y = 0$, which is exactly the condition for $\mathcal{F}$ to be d-solution-bounded.

Finally, if we use the PrincGen algorithm to sample an ideal $\mathfrak{p}$ only, we again obtain a generator $g_\mathfrak{p}$ of norm at most $n^{1.5}$. Applying the algorithm of Lemma 3.1 to the set $\{g_\mathfrak{p}, g_\mathfrak{p} x, \ldots, g_\mathfrak{p} x^{n-1}\}$, we obtain a basis of $\mathfrak{p}$ with Gram-Schmidt norm at most $\gamma_F \cdot n^{1.5}$, which is less than $\nu / \log n$. □

8 Conclusions and Open Problems

We have presented a homomorphic signature scheme that authenticates polynomial functions of bounded degree on signed data.

There are many open problems that remain in this area. First, as we explained in the introduction, we may desire that derived signatures not leak information about the original data set. This privacy property can be achieved for linear functions (e.g. as in [7] and in this paper), but is an open problem for quadratic and higher degree polynomials.

Second, the security of our scheme could be strengthened by removing the random oracle from our construction. All current linearly homomorphic signature schemes use the random oracle to simulate signatures during a chosen message attack. New ideas are needed to eliminate the random oracle while preserving the homomorphic properties.

Third, it is an open problem to base the security of our system on *worst case* problems on ideal lattices. In particular, we wish to generate ideals for our polynomially homomorphic signature scheme from a distribution that admits a random self-reduction. While Gentry [19] has achieved this result for homomorphic encryption, his key generation algorithm is not suitable for our scheme: it produces an ideal $\mathfrak{q}$ and a short vector in $\mathfrak{q}^{-1}$, whereas we require a short vector in $\mathfrak{q}$. One direction for future work is to construct a homomorphic signature scheme that uses Gentry's key generation algorithm; another is to construct an algorithm that samples a uniformly random ideal $\mathfrak{q}$ along with a short vector in $\mathfrak{q}$.

Finally, our construction can be seen as a first step on the road to a *fully homomorphic signature scheme*, which could authenticate the computation of *any* function on signed data. A fully homomorphic signature scheme would be a useful parallel to existing fully homomorphic encryption systems. Current constructions of fully homomorphic encryption are obtained by applying a "bootstrapping" process to a scheme that allows a limited amount of computation on encrypted data. It is unclear whether Gentry's bootstrapping process [18] can be applied to signature schemes such as ours. We leave this as a beautiful open problem. Even if a fully homomorphic scheme cannot be immediately realized, it would be useful to enlarge the set of admissible functions $\mathcal{F}$.

Acknowledgments. The authors thank Rosario Gennaro, Craig Gentry, Hugo Krawczyk, Gil Segev, and Brent Waters for helpful discussions about this work.

References

[1] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, A. Shelat, and B. Waters. "Computing on authenticated data." Cryptology ePrint Archive, Report 2011/096 (2011). http://eprint.iacr.org/.

[2] 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.

[3] 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.

[4] B. Applebaum, Y. Ishai, and E. Kushilevitz. "From secrecy to soundness: Efficient verification via secure computation." In *Proc. of . ICALP* (2010), 152–163.

[5] G. Ateniese, D. H. Chou, B. de Medeiros, and G. Tsudik. "Sanitizable signatures." In *Computer Security — ESORICS '05*, Springer LNCS **3679** (2005), 159–177.

[6] M. Bellare and G. Neven. "Transitive signatures: new schemes and proofs." *IEEE Transactions on Information Theory* **51** (2005), 2133–2151.

[7] D. Boneh and D. Freeman. "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures." In *Public Key Cryptography — PKC '11*, Springer LNCS **6571** (2011), 1–16. Full version avaiable at http://eprint.iacr.org/2010/453.

[8] 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.

[9] C. Brzuska, H. Busch, Ö. Dagdelen, M. Fischlin, M. Franz, S. Katzenbeisser, M. Manulis, C. Onete, A. Peter, B. Poettering, and D. Schröder. "Redactable signatures for tree-structured data: Definitions and constructions." In *Applied Cryptography and Network Security — ACNS '10*, Springer LNCS **6123** (2010), 87–104.

[10] C. Brzuska, M. Fischlin, T. Freudenreich, A. Lehmann, M. Page, J. Schelbert, D. Schröder, and F. Volk. "Security of sanitizable signatures revisited." In *Public Key Cryptography — PKC '09*, Springer LNCS **5443** (2009), 317–336.

[11] E.-C. Chang, C. L. Lim, and J. Xu. "Short redactable signatures using random trees." In *Topics in Cryptology — CT-RSA '09*, Springer LNCS **5473** (2009), 133–147.

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

[13] K.-M. Chung, Y. Kalai, and S. Vadhan. "Improved delegation of computation using fully homomorphic encryption." In *Proc. of Crypto* (2010), 483–501.

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

[15] C. Fragouli and E. Soljanin. "Network coding fundamentals." *Found. Trends Netw.* **2** (2007), 1–133.

[16] R. Gennaro, C. Gentry, and B. Parno. "Non-interactive verifiable computing: Outsourcing computation to untrusted workers." In *Advances in Cryptology — CRYPTO '10*, Springer LNCS **6223** (2010), 465–482.

[17] 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.

[18] C. Gentry. *A fully homomorphic encryption scheme*. Ph.D. dissertation, Stanford University (2009). Available at http://crypto.stanford.edu/craig.

[19] C. Gentry. "Toward basing fully homomorphic encryption on worst-case hardness." In *Advances in Cryptology — CRYPTO '10*, Springer LNCS **6223** (2010), 116–137.

[20] C. Gentry, C. Peikert, and V. Vaikuntanathan. "Trapdoors for hard lattices and new cryptographic constructions." In *40th ACM Symposium on Theory of Computing — STOC '08*. ACM (2008), 197–206.

[21] C. Gentry and D. Wichs. "Separating succinct non-interactive arguments from all falsifiable assumptions." In *Proc. of STOC* (2011). Cryptology ePrint Archive: Report 2010/610.

[22] S. Goldwasser, Y. Kalai, and G. Rothblum. "Delegating computation: Interactive proofs for muggles." In *40th ACM Symposium on Theory of Computing — STOC '08* (2008), 113–122.

[23] S. Haber, Y. Hatano, Y. Honda, W. Horne, K. Miyazaki, T. Sander, S. Tezoku, and D. Yao. "Efficient signature schemes supporting redaction, pseudonymization, and data deidentification." In *ACM Symposium on Information, Computer and Communications Security — ASIACCS '08* (2008), 353–362.

[24] S. Hohenberger. *The cryptographic impact of groups with infeasible inversion*. Master's thesis, MIT (2003).

[25] 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.

[26] 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.

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

[28] V. Lyubashevsky, C. Peikert, and O. Regev. "On ideal lattices and learning with errors over rings." In *Advances in Cryptology — EUROCRYPT '10*, Springer LNCS **6110** (2010), 1–23.

[29] S. Micali. "Computationally sound proofs." *SIAM J. of Computing* **30** (2000), 1253–1298. Extended abstract in FOCS 1994.

[30] S. Micali and R. Rivest. "Transitive signature schemes." In *Proc. of the Cryptographer's Track at the RSA Conference 2002*, LNCS **2271** (2002), 236–243.

[31] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: A cryptographic perspective*, The Kluwer International Series in Engineering and Computer Science **671**. Kluwer Academic Publishers, Boston, Massachusetts (2002).

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

[33] K. Miyazaki, G. Hanaoka, and H. Imai. "Digitally signed document sanitizing scheme based on bilinear maps." In *ACM Symposium on Information, Computer and Communications Security — ASIACCS '06* (2006), 343–354.

[34] K. Miyazaki, M. Iwamura, T. Matsumoto, R. Sasaki, H. Yoshiura, S. Tezuka, and H. Imai. "Digitally signed document sanitizing scheme with disclosure condition control." *IEICE Transactions on Fundamentals* **E88-A** (2005), 239–246.

[35] H. L. Montgomery and R. C. Vaughan. *Multiplicative number theory. I. Classical theory*, Cambridge Studies in Advanced Mathematics **97**. Cambridge University Press, Cambridge (2007).

[36] D. Naccache. "Is theoretical cryptography any good in practice?" CHES 2010 invited talk (2010). Available at www.iacr.org/workshops/ches/ches2010.

[37] J. Quinlan. "Induction of decision trees." *Machine Learning* **1** (1986), 81–106.

[38] J. T. Schwartz. "Fast probabilistic algorithms for verification of polynomial identities." *J. Assoc. Comput. Mach.* **27** (1980), 701–717.

[39] N. P. Smart and F. Vercauteren. "Fully homomorphic encryption with relatively small key and ciphertext sizes." In *Public Key Cryptography — PKC '10*, Springer LNCS **6056** (2010), 420–443.

[40] G. Smith. *Introductory mathematics: Algebra and analysis*. Springer-Verlag, London (1998).

[41] D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. "Efficient public-key encryption based on ideal lattices." In *Advances in Cryptology — ASIACRYPT '09*, Springer LNCS **5912** (2009), 617–635.

[42] R. Steinfeld, L. Bull, and Y. Zheng. "Context extraction signatures." In *Information Security and Cryptology (ICISC)*, Springer LNCS **2288** (2001), 285–304.

[43] P. Stevenhagen. "The arithmetic of number rings." In *Algorithmic number theory: Lattices, number fields, curves and cryptography*, Math. Sci. Res. Inst. Publ. **44**. Cambridge Univ. Press, Cambridge (2008), 209–266.

[44] P. Valiant. "Incrementally verifiable computation or proofs of knowledge imply time/space efficiency." In *Proc. of TCC'08* (2008), 1–18.

[45] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. "Fully homomorphic encryption over the integers." In *Advances in Cryptology — EUROCRYPT '10*, Springer LNCS **6110** (2010), 24–43.

[46] J. Xu. "On directed transitive signature." Cryptology ePrint Archive, Report 2009/209 (2009).

[47] X. Yi. "Directed transitive signature scheme." In *CT-RSA*, LNCS **4377** (2007), 129–144.

[48] 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 Homomorphic Signatures for Linear Functions over Large Fields

We now describe a variant of the linearly homomorphic scheme of Section 4 in which the data can be defined over large fields $\mathbb{F}_p$. In this version, both lattices in the scheme are of the form $\Lambda_q^\perp(\mathbf{A})$ for the same prime q, and thus their intersection is also of this form. We generate a short basis for $\Lambda_1 \cap \Lambda_2$ by generating an $\mathbf{A}$ via TrapGen and using half of its rows to define Λ_1 and the other half to define Λ_2.

This construction allows data to be defined over much larger fields than our first construction, including fields $\mathbb{F}_p$ where p is exponential in n. However, we can only accommodate "small" $\mathbb{F}_p$-linear functions on the data; i.e., functions whose coefficients lift to small integers.

We now give a formal description of the scheme.

$\mathsf{Setup}(1^n, k)$. On input a security parameter n and a maximum data set size k, do the following:

1. Choose a prime $q \geq (nk)^2$. Define $\ell := \lfloor n/6 \log q \rfloor$.
2. Use $\mathsf{TrapGen}(q, 2\ell, 2n)$ to generate a matrix $\mathbf{A} \in \mathbb{F}_q^{2\ell \times 2n}$ along with a short basis $\mathbf{T}$ of $\Lambda_q^\perp(\mathbf{A})$.
3. Let $\mathbf{A}_1 \in \mathbb{F}_q^{\ell \times 2n}$ be the first ℓ rows of $\mathbf{A}$ and $\mathbf{A}_2 \in \mathbb{F}_q^{\ell \times 2n}$ be the last ℓ rows. Define $\Lambda_1 := \Lambda_q^\perp(\mathbf{A}_1)$ and $\Lambda_2 := \Lambda_q^\perp(\mathbf{A}_2)$.
4. Set $\nu := \sqrt{n \log q} \cdot \log n$. Choose an integer $y = \mathrm{poly}(n)$ with $y \leq \sqrt{q}/nk$.
5. Let $H \colon \{0,1\}^* \to \mathbb{F}_q^\ell$ be a hash function (modeled as a random oracle).
6. Output the public key $\mathsf{pk} := (\Lambda_1, \Lambda_2, \nu, k, y, H)$ and the secret key $\mathsf{sk} = \mathbf{T}$.

The public key pk defines the following system parameters:

- The message space is $\mathbb{F}_q^\ell$ and signatures are short vectors in $\mathbb{Z}^{2n}$.
- The set of admissible functions $\mathcal{F}$ is all $\mathbb{F}_q$-linear functions on k-tuples of messages in $\mathbb{F}_q^\ell$ with coefficients in $-y, \ldots, y$.
- For a function $f \in \mathcal{F}$ defined by $f(m_1, \ldots, m_k) = \sum_{i=1}^k c_i m_i$, we encode f by writing the c_i as integers in $(-q/2, q/2]$ and defining $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$.
- To evaluate the hash function ω_τ on an encoded function $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, do the following:
  (a) For $i = 1, \ldots, k$, compute $\alpha_i \leftarrow H(\tau \| i)$.
  (b) Define $\omega_\tau(\langle f \rangle) := \sum_{i=1}^k c_i \alpha_i \ \in \mathbb{F}_q^\ell$.

$\mathsf{Sign}(\mathsf{sk}, \tau, m, i)$. On input a secret key sk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_q^\ell$, and an index i, do:

1. Compute $\alpha_i := H(\tau \| i) \in \mathbb{F}_q^\ell$.
2. Compute $\mathbf{t} \in \mathbb{Z}^n$ such that $\mathbf{A}_1 \cdot \mathbf{t} \bmod q = m$ and $\mathbf{A}_2 \cdot \mathbf{t} \bmod q = \alpha_i$.
3. Output $\sigma \leftarrow \mathsf{SamplePre}(\Lambda_1 \cap \Lambda_2, \mathbf{T}, \mathbf{t}, \nu) \ \in (\Lambda_1 \cap \Lambda_2) + \mathbf{t}$.

$\mathsf{Verify}(\mathsf{pk}, \tau, m, \sigma, f)$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a message $m \in \mathbb{F}_q^\ell$, a signature $\sigma \in \mathbb{Z}^{2n}$, and a function $f \in \mathcal{F}$, do:

1. If all of the following conditions hold, output 1 (accept); otherwise output 0 (reject):
   (a) $\|\sigma\| \leq k \cdot y \cdot \nu\sqrt{n}$.
   (b) $\mathbf{A}_1 \cdot \sigma \bmod q = m$.
   (c) $\mathbf{A}_2 \cdot \sigma \bmod q = \omega_\tau(\langle f \rangle)$.

$\mathsf{Evaluate}(\mathsf{pk}, \tau, f, \vec{\sigma})$. On input a public key pk, a tag $\tau \in \{0,1\}^n$, a function $f \in \mathcal{F}$ encoded as $\langle f \rangle = (c_1, \ldots, c_k) \in \mathbb{Z}^k$, and a tuple of signatures $\sigma_1, \ldots, \sigma_k \in \mathbb{Z}^{2n}$, do:

1. Output $\sum_{i=1}^k c_i \sigma_i$.

Relation to abstract scheme. We now show how the Setup algorithm above instantiates the abstract Setup algorithm of Section 7.1. The numbered items below correspond to the steps of the abstract Setup algorithm.

1. We have $p = q$, $\Lambda_1 = \Lambda_q^\perp(\mathbf{A}_1)$, and $\Lambda_2 = \Lambda_q^\perp(\mathbf{A}_2)$, where $\mathbf{A}_1$ and $\mathbf{A}_2$ are the first ℓ and last ℓ rows of the matrix $\mathbf{A}$ produced by TrapGen.

2. Let $r = \ell$, and define the quotient map $\phi_1 : \mathbb{Z}^{2n} \to \mathbb{F}_q^\ell$ by $\mathbf{v} \mapsto (\mathbf{A}_1 \cdot \mathbf{v} \bmod q)$.
   Let $s = \ell$, and define the quotient map $\phi_2 : \mathbb{Z}^{2n} \to \mathbb{F}_q^\ell$ by $\mathbf{v} \mapsto (\mathbf{A}_2 \cdot \mathbf{v} \bmod q)$.

3. For an integer $k = \ell \leq \sqrt{q}/n$, the sets $\mathcal{F}_0$ and $\mathcal{G}_0$ are the set of projection maps π_i on $(\mathbb{F}_p^\ell)^k$. The set $\mathcal{H}_0$ is the set of projection maps on $(\mathbb{Z}^{2n})^k$. We have $d = 1$ and $|h_i| = 1$ for all i.

4. The set $\mathcal{C}$ is integers in $[-y, y]$.

The parameter λ is equal to $k \cdot y$.

Lemma A.1. *The linearly homomorphic signature scheme defined above is correct with overwhelming probability.*

Proof. This follows from correctness of the abstract scheme (Lemma 7.2). Alternatively, one could imitate the proof of Lemma 4.1. □

Theorem A.2. *If $\mathsf{SIS}_{q,2n,\beta}$ is infeasible for $\beta = 2k \cdot y \cdot n \log n \sqrt{\log q}$, then the homomorphic signature scheme described above is unforgeable.*

Proof. By Theorem 3.2, the distribution of matrices $\mathbf{A}$ generated by $\mathsf{TrapGen}$ is statistically close to uniform over $\mathbb{F}_q^{2\ell \times 2n}$. Since we take the last n rows of $\mathbf{A}$ to define Λ_2, the distribution of challenges for the $\mathcal{L}_2$-$\mathsf{SIS}_{2n,2\lambda\cdot\nu\sqrt{n}}$ problem is statistically close to the distribution of challenges for the $\mathsf{SIS}_{q,2n,2\lambda\cdot\nu\sqrt{n}}$ problem.

We now check the hypotheses of Theorem 7.4. First, note that $\Lambda_1 + \Lambda_2 = \mathbb{Z}^{2n}$ if and only if $\ker \mathbf{A}_1 + \ker \mathbf{A}_2 = \mathbb{F}_q^{2n}$ (here we consider the matrices as maps from $\mathbb{F}_q^{2n}$ to $\mathbb{F}_q^{\ell}$). Since $\ker \mathbf{A}_1$ and $\ker \mathbf{A}_2$ are independent random $(2n-\ell)$-dimensional subspaces of $\mathbb{F}_q^{2n}$ and $\ell < n$, by Lemma A.3 below the probability that $\ker \mathbf{A}_1 + \ker \mathbf{A}_2$ is a proper subspace of $\mathbb{F}_q^{2n}$ is negligible in n.

Next, since $\Lambda_1 \cap \Lambda_2 = \Lambda_q^{\perp}(\mathbf{A})$, by Lemma 3.5 we have $\eta_\epsilon(\Lambda_1 \cap \Lambda_2) < \nu$ for negligible ϵ with overwhelming probability. In addition, the algorithm $\mathsf{TrapGen}$ can be used to sample a (nearly) uniform $\mathbf{A}_1 \in \mathbb{F}_q^{\ell \times 2n}$ along with a basis of length $O(\sqrt{n \log q}) < \nu/\omega(\sqrt{\log n})$. Finally, we observe that for any $\mathbb{F}_q$-vector space V, any nonzero $\mathbb{F}_q$-linear function $f : V^k \to V$ is 1-solution-bounded. □

Lemma A.3. *Let q be a prime, and let A and B be uniformly random matrices in $\mathbb{F}_q^{n\times m}$ and let V_A and V_B be the vector spaces spanned by their rows respectively. If $m < 3n/2$, then the probability that $V_A + V_B = \mathbb{F}_q^m$ is $1 - \mathrm{negl}(n)$.*

Proof. By a standard argument, the probability that $2n > m + (n/2)$ random vectors in $\mathbb{F}_q^m$ span all $\mathbb{F}_q^m$ is $1 - \mathrm{negl}(n)$. Hence, $V_A + V_B$, which is spanned by the $2n$ rows of A and B, will span all of $\mathbb{F}_q^m$ with probability $1 - \mathrm{negl}(n)$. □

Theorem A.4. *Suppose that ν defined in the Setup algorithm satisfies $\nu > y^s \cdot k^s \cdot \omega(\sqrt{\log n})$. Then the linearly homomorphic signature scheme described above is s-weakly context hiding for data sets of size k.*

Proof. The proof is exactly analogous to the proof of Theorem 4.3. The condition $\nu > y^s \cdot k^s \cdot \omega(\sqrt{\log n})$ allows us to conclude that $\nu > \eta_\epsilon(\Lambda_1 \cap \Lambda_2)$ for negligible ϵ with overwhelming probability. This condition can be relaxed if ν is increased and the unforgeability statement is correspondingly weakened; see Remark 4.4. □