Homomorphic Signatures for Polynomial Functions

Dan Boneh[⋆] and David Mandell Freeman[⋆⋆]

Stanford University
{dabo,dfreeman}@cs.stanford.edu

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 [17,33,12], 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).

[⋆] Supported by NSF, DARPA, and AFOSR.

[⋆⋆] Supported by an NSF Mathematical Sciences Postdoctoral Research Fellowship.

The derived signature authenticates both the function f and the result of 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 [23, 11, 38, 7, 16, 6]. 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 [14].

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 [6]. 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.

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 [26, 37, 20] 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 [26] 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 $\big(\tau, \sigma,\ t := f(D),\ \pi\big)$ where π is a short proof that there exists a data set D such that $t = f(D)$ and that 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 by Alice. Security is proved using Valiant's witness extractor [37] 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. 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 eliminates the proof π. The server only publishes $(\tau, \sigma', t := f(D))$, 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 [37], it is much simpler with homomorphic signatures.

More recently Goldwasser, Kalai, and Rothblum [20] and Gennaro, Gentry, and Parno [15] show how to outsource computation securely. In both [15] and [20] (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 [35, 22,21,4,29,28,10,9,8,1,31]. 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 independently 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.

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 $\sigma = \sigma' \mod \Lambda_1$ but $\sigma \neq \sigma' \mod \Lambda_2$. 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 [19] 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 [6].

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 [17]. Ideal lattices also appear in the lattice-based public key encryption schemes of Stehle, Steinfeld, Tanaka, and Xagawa [34] and Lyubashevsky, Peikert, and Regev [25] and in the hash functions of Lyubashevsky and Micciancio [24].

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 [27]. 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 [6], it is not difficult to show that our linearly homomorphic signatures satisfy a privacy property (also defined in [6]) 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 $\mathsf{Evaluate}$ 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 $\mathsf{Evaluate}$ 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 $(\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Evaluate})$ 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.*

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

As in [6] we define privacy for homomorphic signatures using a variation of a definition of Brzuska et al. [9]. 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 concept is similar to that of witness indistinguishability [13], where in our setting we treat the original data set as the witness.

Ahn et al. [1] define a stronger notion of privacy, called *strong context hiding*, 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.

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

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

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 at most $\mu(n) \cdot \log k$ with overwhelming probability.

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

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

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. To handle fractional entries in A we can pre-multiply A by a known scalar to cancel denominators.

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. In the full version of this paper [5] we show that a homomorphic signature scheme for polynomials of degree d^2+d+1 over the integers is sufficient for deriving a signature on the least squares fit. In particular, for a least squares fit using a line it suffices for the signature to be homomorphic for cubic polynomials.

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 [32]). 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 $\mathbb{R}^n$. Standard results on lattices that we use appear in the full version of this paper [5]. Here we note briefly that our schemes will use an algorithm SamplePre [19, Theorem 5.9] that takes as input a basis $\mathbf{T}$ of an n-dimensional lattice Λ, a parameter ν, and a vector $\mathbf{t} \in \mathbb{Z}^n$, and outputs a vector in the coset $\Lambda + \mathbf{t}$ sampled from a Gaussian distribution. SamplePre is itself built from an algorithm SampleGaussian [19, Theorem 4.1] that outputs a vector in the lattice Λ sampled from a Gaussian distribution.

Our linearly homomorphic schemes will use "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 $\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^n : \mathbf{A} \cdot \mathbf{e} = \mathbf{0} \bmod q\}$. Our schemes will use an algorithm TrapGen [3, Theorem 3.2] that samples an (almost) uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$ along with a "short" basis for $\Lambda_q^\perp(\mathbf{A})$.

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 3.1. 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 [27]. 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 [27, §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 [11, 7] or for a small number of vectors [6]. 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. In the full version of this paper [5] 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 [27], 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 $p = \text{poly}(n)$ with $p \le \sqrt{q}/nk$ 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 [19], 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

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

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 $m_1, \ldots, m_k$.

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 $\mathbf{T} := p \cdot \mathbf{T}_q$.
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) $\|\sigma\| \leq k \cdot \frac{p}{2} \cdot \nu\sqrt{n}$.
   (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 $\sigma := \sum_{i=1}^{k} c_i \sigma_i$.

In the full version of this paper [5], we show that this linearly homomorphic signature scheme is correct with overwhelming probability and that it is length efficient; i.e., the bit length of a derived signature depends logarithmically on the data set size k.

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 [3, 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 4.1. *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.*

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 a Gaussian distribution on $\Lambda_1 + m$ 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. 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 [19, 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 [6] 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 $k = \text{poly}(n)$ vectors per data set.

Privacy. We now show that our linearly homomorphic signature scheme is weakly context hiding. Specifically, we 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. The proof of the following theorem can be found in the full version of this paper [5].

Theorem 4.2. *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 k.*

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 $\{1, x, x^2, \dots, x^{\deg f - 1}\}$ 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 $\gamma_f \le \sqrt{n}$ (cf. [17, 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 [36]; 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 $\mathcal{O}_K$.

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 [34, 24], 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 [36, 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 $\mathbb{Z}[x]/(f(x)) \rightarrow \mathbb{F}_{p^e}$; 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 [33]. 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 ([33, §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 [30, 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 let $\mathcal{O}_K$ be the lattice in $\mathbb{Q}^n$ corresponding (via the coefficient embedding) to the ring of integers of K. We now let Λ_1 and Λ_2 be (degree one) prime ideals $\mathfrak{p}, \mathfrak{q} \subset \mathcal{O}_K$ of norm p, q respectively. We fix an isomorphism from $\mathcal{O}_K/\mathfrak{p}$ to $\mathbb{F}_p$ by representing $\mathfrak{p}$ as $p\mathcal{O}_K + (x-a)\mathcal{O}_K$ and mapping $h(x) \in \mathcal{O}_K$ to $h(a) \bmod p \in \mathbb{F}_p$, and similarly for $\mathcal{O}_K/\mathfrak{q} \cong \mathbb{F}_q$. 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 $f \bmod p$. 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 $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{O}_K$, 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 $\text{poly}(n)$ produces a vector of length $\text{poly}(n)$. 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 $\gamma_F = \text{poly}(n)$.
   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 $\mathsf{PrincGen}$ 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 $y = \text{poly}(n)$ 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. 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 $\hat{f}(\sigma_1, \ldots, \sigma_k)$.

In the full version of this paper [5], we show that this polynomially homomorphic signature scheme is correct with overwhelming probability and that it is length efficient; i.e., the bit length of a derived signature depends logarithmically on the data set size k.

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 6.1. *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*

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

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

The proof of this theorem uses the same ideas as that of Theorem 4.1; details are in the full version of this paper [5]. While Theorem 6.1 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.

Privacy. The homomorphic signature scheme described above is not weakly context hiding in the sense of Definition 2.3. To see why, consider an attacker that outputs two data sets $\vec{m}_0^* := (0,0)$ and $\vec{m}_1^* := (0,1)$, 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 siganture $\sigma = \sigma_1\sigma_2$ 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 $\Pr[\sigma \in \mathfrak{p}^2] \le 1/2$ 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.

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 the full version [5], we show how to authenticate polynomial functions of data defined over a field where p is constant or polynomial in n, as we can for linear computations using the scheme of Section 4.

7 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 [6] 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 [18] 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 [17] 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. Ahn, J.H., Boneh, D., Camenisch, J., Hohenberger, S., Shelat, A., Waters, B.: Computing on authenticated data (2010) (manuscript)
2. Ajtai, M.: Generating hard instances of the short basis problem. In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 1–9. Springer, Heidelberg (1999)
3. Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS, pp. 75–86 (2009), full version `http://www.cc.gatech.edu/~cpeikert/pubs/shorter.pdf`
4. Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable signatures. In: di Vimercati, S.D.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 159–177. Springer, Heidelberg (2005)
5. Boneh, D., Freeman, D.: Homomorphic signatures for polynomial functions. Cryptology ePrint Archive, report 2011/018 (2011), `http://eprint.iacr.org/2011/018`
6. Boneh, D., Freeman, D.: Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In: Gennaro, R. (ed.) PKC 2011. LNCS, vol. 6571, pp. 1–16. Springer, Heidelberg (2011), full version `http://eprint.iacr.org/2010/453`

7. Boneh, D., Freeman, D., Katz, J., Waters, B.: Signing a linear subspace: Signature schemes for network coding. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 68–87. Springer, Heidelberg (2009)
8. Brzuska, C., Busch, H., Dagdelen, Ö., Fischlin, M., Franz, M., Katzenbeisser, S., Manulis, M., Onete, C., Peter, A., Poettering, B., Schröder, D.: Redactable signatures for tree-structured data: Definitions and constructions. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 87–104. Springer, Heidelberg (2010)
9. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schröder, D., Volk, F.: Security of sanitizable signatures revisited. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 317–336. Springer, Heidelberg (2009)
10. Chang, E.C., Lim, C.L., Xu, J.: Short redactable signatures using random trees. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 133–147. Springer, Heidelberg (2009)
11. Charles, D., Jain, K., Lauter, K.: Signatures for network coding. International Journal of Information and Coding Theory 1(1), 3–14 (2009)
12. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)
13. Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. In: Proc. of the 22nd ACM Symposium on Theory of Computing, pp. 416–426 (1990)
14. Fragouli, C., Soljanin, E.: Network coding fundamentals. Found. Trends Netw. 2(1), 1–133 (2007)
15. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 465–482. Springer, Heidelberg (2010)
16. Gennaro, R., Katz, J., Krawczyk, H., Rabin, T.: Secure network coding over the integers. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 142–160. Springer, Heidelberg (2010)
17. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009), http://crypto.stanford.edu/craig
18. Gentry, C.: Toward basing fully homomorphic encryption on worst-case hardness. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 116–137. Springer, Heidelberg (2010)
19. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: 40th ACM Symposium on Theory of Computing—STOC 2008, pp. 197–206. ACM, New York (2008)
20. Goldwasser, S., Kalai, Y., Rothblum, G.: Delegating computation: Interactive proofs for muggles. In: 40th ACM Symposium on Theory of Computing — STOC 2008, pp. 113–122 (2008)
21. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., Yao, D.: Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In: ASIACCS 2008, pp. 353–362. ACM, New York (2008)
22. Johnson, R., Molnar, D., Song, D., Wagner, D.: Homomorphic signature schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (2002)
23. Krohn, M., Freedman, M., Mazieres, D.: On-the-fly verification of rateless erasure codes for efficient content distribution. In: Proc. of IEEE Symposium on Security and Privacy, pp. 226–240 (2004)
24. Lyubashevsky, V., Micciancio, D.: Generalized compact knapsacks are collision resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 144–155. Springer, Heidelberg (2006)
25. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010)

26. Micali, S.: Computationally sound proofs. SIAM J. of Computing 30(4), 1253–1298 (2000); extended abstract in FOCS 1994
27. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. In: 45th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2004, pp. 372–381. IEEE Computer Society, Washington, DC, USA (2004)
28. Miyazaki, K., Hanaoka, G., Imai, H.: Digitally signed document sanitizing scheme based on bilinear maps. In: ACM Symposium on Information, Computer and Communications Security — ASIACCS 2006, pp. 343–354 (2006)
29. Miyazaki, K., Iwamura, M., Matsumoto, T., Sasaki, R., Yoshiura, H., Tezuka, S., Imai, H.: Digitally signed document sanitizing scheme with disclosure condition control. IEICE Transactions on Fundamentals E88-A(1), 239–246 (2005)
30. Montgomery, H.L., Vaughan, R.C.: Multiplicative number theory. I. Classical theory. Cambridge Studies in Advanced Mathematics, vol. 97. Cambridge University Press, Cambridge (2007)
31. Naccache, D.: Is theoretical cryptography any good in practice? CHES 2010 invited talk (2010), `http://www.iacr.org/workshops/ches/ches2010`
32. Quinlan, J.: Induction of decision trees. Machine Learning 1, 81–106 (1986)
33. Smart, N.P., Vercauteren, F.: Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 420–443. Springer, Heidelberg (2010)
34. Stehlé, D., Steinfeld, R., Tanaka, K., Xagawa, K.: Efficient public key encryption based on ideal lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 617–635. Springer, Heidelberg (2009)
35. Steinfeld, R., Bull, L., Zheng, Y.: Content extraction signatures. In: Kim, K. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 285–304. Springer, Heidelberg (2002)
36. Stevenhagen, P.: The arithmetic of number rings. In: Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography. Math. Sci. Res. Inst. Publ., vol. 44, pp. 209–266. Cambridge Univ. Press, Cambridge (2008)
37. Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 1–18. Springer, Heidelberg (2008)
38. Zhao, F., Kalker, T., Médard, M., Han, K.: Signatures for content distribution with network coding. In: Proc. Intl. Symp. Info. Theory (ISIT) (2007)