Efficient Commitments and Zero-Knowledge Protocols from Ring-SIS with Applications to Lattice-based Threshold Cryptosystems

Carsten Baum¹, Ivan Damgård¹, Sabine Oechsner¹, and Chris Peikert²

¹ Department of Computer Science, Aarhus University {cbaum, ivan, oechsner}@cs.au.dk
² Department of Computer Science and Engineering, University of Michigan cpeikert@umich.edu

Abstract. We present an additively homomorphic commitment scheme with hardness based on the Ring-SIS problem. Our construction is statistically hiding as well as computationally binding and allows to commit to a vector of ring elements at once.

We show how to instantiate efficient zero-knowledge protocols that can be used to prove a number of relations among these commitments, and apply these in the context of lattice-based threshold cryptosystems: we give a generic transformation that can be used with certain (Ring-)LWEbased encryption schemes to make their algorithms actively secure. We show how this transformation can be used to implement distributed decryption with malicious security as well as maliciously secure threshold key generation in an efficient way.

1 Introduction

Since Regev [Reg05] introduced the LWE problem and demonstrated its connection to lattice problems, lattice-based cryptography has developed and matured rapidly. As its development continues, it becomes more and more important to have a full set of efficient tools and protocols based on the hardness of it. This is particularly necessary as it currently is one of the promising candidates as a potential post-quantum replacement for the discrete logarithm problem and factoring. Therefore, it is of importance to construct standard cryptographic primitives such as encryption or commitment schemes based on it, plus companion protocols such as zero-knowledge proofs or threshold key generation.

Commitment schemes [Blu82] are a key tool in the design of cryptographic protocols and have countless applications. In particular, when combined with zero-knowledge proofs, they allow to enforce "good" behavior by adversarial parties and make the design of protocols secure against malicious attacks easier. A prime example where these can be used is in the context of latticebased threshold encryption schemes [DF89], where multiple parties collaborate to generate a public/private key pair or decrypt ciphertexts using an interactive protocol. Such encryption schemes are applied e.g. in the SPDZ MPC protocol [DPSZ12,DKL⁺13], where Damgård et al. used a variant of the cryptosystem from [BV11,BGV12] in their preprocessing protocol. However, in [DPSZ12], the question of key generation was avoided by assuming that a common public key and shares of the corresponding secret key were already in place (due to the lack of an efficient, actively secure protocol). [DKL⁺13] gave a key generation procedure, but it was only covertly secure. As for distributed decryption, both papers used a simple semi-honestly secure solution: decryption might produce an incorrect output when players cheat, and the surrounding protocol has to check this output and catch any errors later. This error checking leads to additional overhead in both computation and communication, which is unsatisfying.

This clearly leaves several problems open: first, we would like to have maliciously secure distributed key generation protocols that are efficient and avoid the use of generic zero-knowledge techniques. Second, we would like to have distributed key generation protocols that do not have to rely on external checks for malicious security.

In this paper we will therefore be interested in constructing commitment schemes based on lattice-related problems such as the Ring-SIS problem and efficient zero-knowledge protocols for proving relations on committed values. Loosely speaking, in the Ring-SIS problem, one is given a number of elements a_1, \ldots, a_m in a ring and the goal is to find ring elements x_1, \ldots, x_m such that $x_1a_1 + \cdots + x_ma_m = 0$ under the constraint that the x_i 's must be "small" (in some appropriate sense that we explain in more detail later).

There are several earlier works in this area: Kawachi et al.'s work about identification schemes [KTX08] presents a string commitment scheme based on the SIS assumption, where one commits to vectors over \mathbb{Z}_q . However, the message space is restricted to vectors of small norm; otherwise, the binding property is lost. This restriction causes problems in the applications we are interested in: for instance, if a player wants to prove (efficiently) that he has performed an encryption or decryption operation correctly in a cryptosystem that uses the ring \mathbb{Z}_q , one typically requires a commitment scheme that is linearly homomorphic and can commit to *arbitrary* vectors over \mathbb{Z}_q rather that only short ones.

In [JKPT12], Jain et al. proposed a commitment scheme where the hiding property is based on the Learning Parity with Noise (LPN) assumption, a variant of the LWE assumption. They also constructed zero-knowledge proofs to prove general relations on bit strings. A generalization of [JKPT12] was proposed by Xie et al. [XXW13]. Their work presents a commitment scheme that is based on Ring-LWE instead of LPN, and they build Σ -protocols from it. Further Σ protocols based on (Ring-)LWE encryption schemes were presented by Asharov et al. [AJL⁺12] and Benhamouda et al. [BCK⁺14].

A main problem with all these previous schemes is that the zero-knowledge proofs had a non-negligible soundness error and hence one needs many iterations to have full security.

However, in [BKLP15], a commitment scheme, as well as companion zeroknowledge protocols were constructed with much better efficiency: one can commit to a vector over finite field \mathbb{F}_q resulting in a commitment that is only a constant factor larger than the committed vector. Furthermore, they gave protocols for proving knowledge of a committed string as well as proving linear and multiplicative relations on committed values. These are efficient in the sense that the soundness error is negligible already for a single iteration of the protocol. The commitments are unconditionally binding and computationally hiding, and the underlying assumption is Ring-LWE over $\mathbb{F}_q[x]/(x^N+1)$, where q is a prime congruent to 3 modulo 8 and N is a power of two.

In [BKLP15], it was not worked out in detail how this might be used for key generation or decryption, so we cannot comment on the exact efficiency one would get from using these tools, but it is reasonable to expect that this is possible and that one could do much better than with the previous "generation" of commitments and protocols. On the other hand, there are also some natural open questions:

First, can we also get the other flavor of commitments, that is, statistical hiding and computational binding and second, since [BKLP15] uses a rather specialized ring, can we get more general results and also perhaps be able to use other lattice-related assumptions? A natural candidate here is the Ring-SIS problem as mentioned above.

1.1 Our Contributions

In this paper, we make the following contributions (where $R_q = \mathbb{F}_q[x]/(f(x))$ with f(x) of degree N).

- 1. We propose a commitment scheme that allows to commit to an N-vector over \mathbb{F}_q , or equivalently, an element in R_q . A commitment consists of two elements from R_q . The scheme is statistically hiding and computationally binding if the Ring-SIS problem over R_q is hard. The scheme works for a class of rings that include the ones from [BKLP15] as a special case. Furthermore, the scheme can be extended to allow commitments to a vector of elements from R_q .
- 2. We give (honest-verifier) zero-knowledge protocols for proving knowledge of committed values, and for linear relations on committed values. The protocols achieve negligible soundness error in one iteration, and the communication overhead for doing a proof (compared to just sending the commitment) is only a logarithmic factor (in the security parameter). Finally, we give a protocol for proving that a committed value is short. This one does not have negligible soundness error, but it can be made efficient in an amortized sense using recent work by Baum et al. [BDLN16]. We note that [BKLP15] also did not have a one-shot efficient shortness proof.
- 3. We show how to use these tools to build maliciously secure threshold key generation and decryption protocols for a class of LWE-based cryptosystems. The communication overhead of the distributed decryption (compared to just sending the ciphertext) is a logarithmic factor (in the security parameter), when the cost is amortized over several decryptions. The amortization comes in as follows: the parties need to generate some special committed values that do not depend on the actual ciphertexts to decrypt later. The cost of this is

cheap when amortized over many values. But once this is done, the parties can decrypt a number of ciphertexts, and each such decryption will be cheap, even if they must be done one at a time.

In comparison to [BKLP15], the most relevant previous work, we have achieved the other flavor of commitments with similar efficiency but smaller expansion factor (2 versus at least 3). Our underlying computational problem is different (Ring-SIS versus Ring-LWE). Moreover, we have a more general class of rings we can work with, which potentially weakens the assumption we have to make it is enough that one of the families of rings in our class will work.

On the technical side, we make use of the observation that the function defining the SIS problem can be seen both as a collision intractable hash function (if the SIS problem is hard) and as a universal hash function (if the underlying ring is chosen appropriately). The committer therefore chooses a random input r and gives away the image of r under one instance of the function so reffectively is fixed. And then he also applies another instance of the function to extract randomness that is used as a one-time pad on the message to commit to. On a very high level this idea goes back to [DPP93]. What distinguishes our construction is that the same type of SIS function can play both roles and this gives nice algebraic properties for the commitment scheme and hence efficient zero-knowledge protocols. We also borrow an idea from [BKLP15] of relaxing the condition for valid opening of a commitment, in return for a better soundness error probability in our protocols.

2 Preliminaries

In this paper, we will make use of the rings $R = \mathbb{Z}[x]/(f(x))$ and $R_q = R/qR = \mathbb{F}_q[x]/(f(x))$, where $f(x) \in \mathbb{Z}[x]$ is a monic irreducible polynomial of degree N, and q is a prime integer. We will base security on the Ring-SIS problem (defined below) for these parameters, and note that the choice of f(x) may affect how hard the problem is. We discuss later concrete suitable choices of f(x).

We identify R with its set of standard representatives, namely, integer polynomials of degree less than N. This leads to the standard "coefficient" norm, defined as $||r_0 + r_1x + \cdots + r_{N-1}x^{N-1}||_{\infty} = \max_i\{|r_i|\}$.

We extend this norm to vectors over R in the usual way: $||(r_1, \ldots, r_m)||_{\infty} = \max_i \{||r_i||_{\infty}\}.$

We define deg(r), the degree of an element in $r \in R$ to be the degree of its minimal degree representative in $\mathbb{Z}[x]$. We extend the definition of degree to vectors as follows: for $\mathbf{r} = (r_1, \ldots, r_m) \in \mathbb{R}^m$, deg(\mathbf{r}) = max_i{deg(r_i)}.

We note that we can make R_q be an *R*-module in a natural way (using the fact that \mathbb{F}_q is a \mathbb{Z} -module), so for any $\mathbf{a} \in R, b \in R_q$, $\mathbf{a} \cdot b$ is in R_q .

Definition 1. The ring $R_q = R/qR$ is $(\mathsf{D}, \gamma_{\mathsf{D}}, d_R)$ -commitment friendly if there is a subset $\mathsf{D} \subseteq R$ such that for any $\mathsf{c} \in \mathsf{D}$ and $r \in R$ with $\deg(r) < 2d_R$, we have $||\mathsf{c} \cdot r||_{\infty} \leq \gamma_{\mathsf{D}}||r||_{\infty}$. Furthermore, for any $\mathsf{d}, \mathsf{d}' \in \mathsf{D}$ it holds that $\mathsf{d} - \mathsf{d}'$ is invertible modulo q, and $\deg(\mathsf{d}) < d_R$. Looking ahead, we will use values in D as challenges sent by the verifier in our protocols, and we will choose randomness for our commitments using $r \in R$ with small norm $||r||_{\infty}$ and $\deg(r) < d_R$. It will be important that multiplying such an element by a challenge (even twice, it turns out) does not enlarge its norm by too much. By limiting the degrees, we avoid reductions modulo f(x), which further controls the norms of the products.

We will assume throughout that one can efficiently check membership in D and invert (nonzero) differences modulo q. For convenience, we will write elements from D with sans-serif throughout, e.g., $f \in D$.

2.1 The Ring-SIS Problem

In the Ring-SIS problem parameterized by a ring R, integer modulus q, and norm bound u < q, one is given a uniformly random vector $\boldsymbol{a} = (a_1, \ldots, a_m) \in R_q^m$ as input, and the goal is to find a nonzero vector $\boldsymbol{s} \in R^m$ such that $\boldsymbol{a} \cdot \boldsymbol{s} = 0 \in R_q$ and $||\boldsymbol{s}||_{\infty} \leq u$.

Whether this is hard depends of course on the parameters and clearly gets no harder as u increases. But the choice of ring R is also important; in particular, the polynomial used for constructing R matters.

One can think of \boldsymbol{a} as the specification of a hash function which is collision intractable if the Ring-SIS instance \boldsymbol{a} is hard. Specifically, the function sends an input vector $\boldsymbol{v} \in R^m$ of norm at most u/2 to $\boldsymbol{a} \cdot \boldsymbol{v} \in R_q$. For any collision, the difference between the colliding inputs trivially gives a Ring-SIS solution.

3 Commitment Scheme

We start by some intuition: in [DPP93], a simple construction of a commitment scheme was proposed based on any collision intractable hash function h. To commit to a bit string x, one chooses a random (sufficiently long) string r, and the commitment is defined as $(h(r), \phi, \phi(r) \oplus x)$, where ϕ is a universal hash function. The idea was that by collision intractability, the committer cannot change his mind about r and hence not about x either. It is hiding by the randomness extraction property of ϕ : if r is long enough compared to h(r), then $\phi(r)$ is essentially uniform and masks x.

One can now observe that the "Ring-SIS function" defined by $\boldsymbol{a} \in R_q^m$ that sends a short vector \boldsymbol{r} to $\boldsymbol{a} \cdot \boldsymbol{r}$ can be thought of as a collision intractable function. However, the same type of function can also be used as a randomness extractor by choosing suitable parameters. The intuition behind our scheme is therefore to instantiate the idea from [DPP93] using instances of the Ring-SIS function over R_q for both h and ϕ .

However, in contrast to standard instantiations of [DPP93], both functions are defined over the same polynomial ring and this gives the scheme some nice algebraic properties. It turns out that we can use these for constructing efficient zero-knowledge protocols for the scheme. We can now describe the commitment scheme we propose in its most general form. Fig. 1 gives an overview of the parameters of the scheme. We leave open the concrete choice of the underlying ring as well as the distribution with which the randomness is chosen. In Section 5, we make suggestions for these and argue that computational binding is likely to hold.

- KeyGen: The public commitment key is the specification of a $(\mathsf{D}, \gamma_{\mathsf{D}}, d_R)$ commitment friendly ring R_q , a uniformly random matrix $A \in R_q^{2 \times m}$ and
 constants $\gamma > 1$ and β such that $\gamma N m \beta \gamma_{\mathsf{D}}^2 < q$. Finally, we assume that a
 distribution \mathcal{D} is given that will output vectors in R^m . It is required that any
 vector that can be produced with non-zero probability has norm at most β and degree less than d_R (we specify this distribution in Section 5).
- Commit: To commit to a message $x \in R_q$, draw a $\mathbf{r} \in R^m$, according to \mathcal{D} (so $||\mathbf{r}||_{\infty} \leq \beta$ and deg $(\mathbf{r}) < d_R$), and compute

$$Com(x; \mathbf{r}) := A\mathbf{r} + \begin{pmatrix} 0 \\ x \end{pmatrix}.$$

- Open: A valid opening of a commitment c is a 3-tuple: $x \in R_q$, $r \in R^m$, and $f \in D$. The verifier checks that

$$A\boldsymbol{r} + \begin{pmatrix} 0 \\ fx \end{pmatrix} = f\boldsymbol{c}$$

and that $||\mathbf{r}||_{\infty} \leq \gamma N m \beta \gamma_{\mathsf{D}}, \deg(\mathbf{r}) < 2d_R.$

We will often omit the choice of randomness and write C(x) or $C(x; \mathbf{r})$ instead of $Com(x; \mathbf{r})$.

Note that an honest committer can always open by letting f = 1, and would always have its value of r be shorter than $\gamma Nm\beta\gamma_D$, namely it would have norm at most β . We only allow for these relaxed conditions in order to get soundness and zero-knowledge for the protocols we propose in Section 4: we will only be able to guarantee that a dishonest prover can open his values using f-values that are (possibly) not 1, and r-values that are somewhat longer than β . This is fine, as long as the scheme is still binding under the relaxed condition, as indeed we show below.

3.1 Security

We will now show properties of our commitment scheme. The first lemma shows that breaking the binding property implies one can solve the Ring-SIS problem over R_q . The second lemma shows that the commitment scheme is statistically hiding.

Lemma 1 (Binding Property). From a commitment c and correct openings r, f, r', f' to two different messages x, x', one can efficiently compute a solution s with $||s||_{\infty} \leq 2Nm\gamma\beta\gamma_{D}^{2}$ to the Ring-SIS problem instance defined by the top row of A.

Parameter	Explanation
f(x)	Monic irreducible integer polynomial defining $R = \mathbb{Z}[x]/(f(x))$
q	Prime integer defining $R_q = R/qR = \mathbb{F}_q[x]/(f(x))$
N	Degree of f
D	Special subset of R , used for challenges
γ_{D}	Norm expansion factor when multiplying by a challenge
d_R	Max degree of ring elements used for randomness in proofs
A	Public matrix over R_q
γ	Constant that regulates the abort probability
\mathcal{D}	Distribution of honest prover's randomness for commitments
β	Norm bound for honest prover's randomness
m	Dimension (over R_q) for the SIS problem
λ, κ	Computational/statistical security parameter

Fig. 1. Overview of Parameters.

Proof. Let c and x, r, f and x', r', f' be as assumed in the lemma. Then

$$A(f'\boldsymbol{r}) + \begin{pmatrix} 0\\ ff'x \end{pmatrix} = ff'\boldsymbol{c} = A(f\boldsymbol{r}') + \begin{pmatrix} 0\\ ff'x' \end{pmatrix}$$

and so

$$A(f'\boldsymbol{r} - f\boldsymbol{r}') = \begin{pmatrix} 0\\ ff'(x-x') \end{pmatrix}.$$

Since $x - x' \neq 0$ and the actions of both f, f' are invertible, we have $ff'(x - x') \neq 0$. Then it must be that also $f'\mathbf{r} - f\mathbf{r'} \neq 0$ since otherwise the above equation would be false. Hence we have found a solution $f'\mathbf{r} - f\mathbf{r'}$ to the Ring-SIS problem instance defined by the top row of A. Furthermore, by the definition of a commitment friendly ring, the norms of $f'\mathbf{r}$ and $f\mathbf{r'}$ are at most $(Nm\gamma\beta\gamma_{\rm D})\cdot\gamma_{\rm D}$ and hence $f'\mathbf{r} - f\mathbf{r'}$ has norm at most $2Nm\gamma\beta\gamma_{\rm D}^2$ by the triangle inequality. \Box

Lemma 2 (Hiding Property). Assume the distribution \mathcal{D} and that R_q , m are chosen such that 1) the min-entropy of a vector drawn from \mathcal{D} is at least $2\log |R_q| + \kappa$ where κ is a (statistical) security parameter, and 2) the class of functions $\{f_a \mid a \in R_q^m\}$ where $f_a(\mathbf{r}) = \mathbf{a} \cdot \mathbf{r}$ is universal when mapping the support of \mathcal{D} to R_q . Then the scheme is statistically hiding.

Proof. Note that a commitment gives the adversary $\log |R_q|$ bits of information on \boldsymbol{r} , namely the dot product of \boldsymbol{r} with the top row \boldsymbol{a}_0 of A. So even given this dot-product we have $\log |R_q| + \kappa$ bits of randomness left in \boldsymbol{r} . Let \boldsymbol{a}_1 be the bottom row of A. Then from the assumptions and the left-over hash lemma, it follows that $f_{\boldsymbol{a}_1}(\boldsymbol{r})$ is statistically close to random, even given $\boldsymbol{a}_0 \cdot \boldsymbol{r}$ and so the scheme is indeed statistically hiding. \Box

Later, when we propose concrete instantiations of R_q in Section 5, we will have to argue that the condition in the lemma is in fact satisfied.

3.2 Extension to message vectors

The commitment scheme can be extended to allow committing to vectors $\boldsymbol{x} \in R_q^k$ (for constant k) by modifying it in the following way:

- KeyGen: Output a (D, γ_D, d_R) -commitment friendly ring R_q , a uniformly random matrix $A \in R_q^{(k+1) \times m}$, constants $\gamma > 1$ and β such that $\gamma Nm\beta\gamma_D^2 < q$, and a distribution \mathcal{D} that will output vectors in R^m . The same restrictions on \mathcal{D} as before apply.
- Commit: To commit to a message $\boldsymbol{x} \in R_q^k$, draw a $\boldsymbol{r} \in R^m$, according to \mathcal{D} (so $||\boldsymbol{r}||_{\infty} \leq \beta$ and deg $(\boldsymbol{r}) < d_R$), and compute

$$Com(\boldsymbol{x};\boldsymbol{r}) := A\boldsymbol{r} + \begin{pmatrix} 0 \\ \boldsymbol{x} \end{pmatrix}.$$

- Open: A valid opening of a commitment c is a 3-tuple: $x \in R_q^k$, $r \in R^m$, and $f \in D$. The verifier checks that

$$A\boldsymbol{r} + \begin{pmatrix} 0 \\ f\boldsymbol{x} \end{pmatrix} = f\boldsymbol{c},$$

and that $||\mathbf{r}||_{\infty} \leq \gamma N m \beta \gamma_{\mathsf{D}}, \deg(\mathbf{r}) < 2d_R.$

The security properties of the extended commitment scheme remain similar:

Lemma 3 (Binding Property). From a commitment c and correct openings r, f, r', f' to two different message vectors x, x', one can efficiently compute a solution s with $||s||_{\infty} \leq 2Nm\gamma\beta\gamma_{D}^{2}$ to the Ring-SIS problem instance defined by the top row of A.

Lemma 4 (Hiding Property). Assume the distribution \mathcal{D} and that R_q , m are chosen such that 1) the min-entropy of a vector drawn from \mathcal{D} is at least $(k+1)\log|R_q| + \kappa$ where κ is a (statistical) security parameter, and 2) the class of functions $\{f_a \mid a \in R_q^m\}$ where $f_a(\mathbf{r}) = \mathbf{a} \cdot \mathbf{r}$ is universal when mapping the support of \mathcal{D} to R_q . Then the scheme is statistically hiding.

The binding and hiding property follow by the same arguments as in Lemma 1 and Lemma 2, respectively.

For the sake of readability, we will use the commitment scheme throughout this paper only to commit to single ring elements $x \in R_q$.

4 Zero-Knowledge Proofs

In this section, we describe Σ -protocols that can be constructed for our commitment scheme. The protocols use rejection sampling as introduced by Lyubashevsky [Lyu08,Lyu09] to hide the randomness of the public commitment. Furthermore, the protocols use an auxiliary commitment scheme C_{aux} to hide the content of the first message from the verifier and thereby enforce the right challenge distribution. We can use our own commitment scheme for this, but we still use the notation C_{aux} to make the presentation clearer. Our protocols will have slightly weaker properties than usually considered for Σ -protocols, but this does not affect their usefulness in practice: We get statistical honest verifier zero-knowledge rather than perfect, and we get computational soundness rather than perfect, in the sense that a prover who can answer two different challenges must either know the witness we want him to know, or he can break the binding property of C_{aux} . All protocols, except the one for proving bounds (Π_{BOUND}), have soundness error $1/|\mathsf{D}|$, which will turn out to be negligible for the instantiations we give in Section 5. Moreover, for all our instantiations, the communication complexity is dominated by the size of the prover's last message, which will be $O(N \log(q) \log(N \log q))$ bits. Note that a commitment is of size $O(N \log q)$ bits, so doing the protocols adds very little asymptotic overhead.

In Section 4.6 we discuss ways in which we can make our protocols be zeroknowledge against a dishonest verifier. Note that this is not obvious (even using rewinding), when the challenge space is large.

4.1 **Proof for Opening a Commitment**

Suppose the prover has published $\boldsymbol{c} = C(x; \boldsymbol{r})$ and claims to know a valid opening. Then consider the following protocol to prove this:

Protocol Π_{Open}

- 1. The prover computes a commitment $\mathbf{t} = C(\mu; \boldsymbol{\rho})$, where μ and $\boldsymbol{\rho}$ are chosen uniformly from R_q and R^m , respectively, subject to $||\boldsymbol{\rho}||_{\infty} \leq (1 + (\gamma Nm)/2)\beta\gamma_{\mathsf{D}}$ and $\deg(\boldsymbol{\rho}) < 2d_R$, and sends $\mathbf{c}_{aux} = C_{aux}(\mathbf{t})$ to the verifier.
- 2. The verifier sends a random challenge $d \in D$.
- 3. The prover first checks that d is a valid challenge. The prover's goal is then to open $\mathbf{t} + \mathbf{dc}$ to $z = \mu + \mathbf{dx}$, $\mathbf{r}_z = \boldsymbol{\rho} + \mathbf{dr}$ (it is understood that $\mathbf{f} = 1$). The protocol is aborted if $||\mathbf{r}_z||_{\infty} > (\gamma Nm)/2 \cdot \beta \gamma_{\mathsf{D}}$. Otherwise, the prover sends to the verifier z, \mathbf{r}_z , and opening information u_{aux} for the commitment \mathbf{c}_{aux} .
- 4. The verifier checks that u_{aux} is valid, that $C(z; \boldsymbol{z}_r) = \boldsymbol{t} + d\boldsymbol{c}$, that $||\boldsymbol{r}_z||_{\infty} \leq (\gamma Nm)/2 \cdot \beta \gamma_{\mathsf{D}}$ and that $\deg(\boldsymbol{r}_z) < 2d_R$.

We now look at the properties of this protocol.

Lemma 5 (Completeness). The verifier always accepts an honest prover when Π_{OPEN} does not abort. The probability of abort is at most $2/\gamma$.

Proof. An honest prover can clearly answer correctly for any challenge d and hence an honest verifier will always accept if the protocol did not abort. Recall that an element in R_q can be thought of as a polynomial with at most N coefficients and the vector \mathbf{r}_z is therefore defined by a total of at most Nm coefficients. Since R_q is commitment friendly, coefficients in $d\mathbf{r}$ have norm at most $\beta\gamma_{\rm D}$, the probability that a single coefficient will cause an abort is

$$p = \frac{2\beta\gamma_{\mathsf{D}}}{2(1+\gamma Nm/2)\beta\gamma_{\mathsf{D}}+1} \le \frac{2}{\gamma Nm} \tag{1}$$

and hence the probability that some coefficient causes an abort is at most $2/\gamma$ by the union bound.

Lemma 6 (Special Soundness). On input commitment \mathbf{c} and a pair of transcripts for Π_{OPEN} ($\mathbf{c}_{aux}, \mathsf{d}, (u_{aux}, t, z, \mathbf{r}_z)$), ($\mathbf{c}_{aux}, \mathsf{d}', (u'_{aux}, t', z', \mathbf{r}'_z)$) where $\mathsf{d} \neq \mathsf{d}'$, we can extract either a witness for breaking the auxiliary commitment scheme, or a valid opening of \mathbf{c} .

Proof. We first note that if $t \neq t'$ in the input information, this breaks binding for the auxiliary scheme (of course one expects that this occurs with negligible probability). Otherwise t = t', and one can compute the message contained in c as $x = f^{-1}(z - z')$ where f = d - d' and is indeed invertible since R_q is commitment-friendly. We set the randomness to be $\mathbf{r} = \mathbf{r}_z - \mathbf{r}'_z$.

This works since if we subtract the two equations the verifier would check in the two transcripts, we obtain

$$(\mathsf{d}-\mathsf{d}')\boldsymbol{c} = A(\boldsymbol{r}_z - \boldsymbol{r}'_z) + \begin{pmatrix} 0\\ z-z' \end{pmatrix},$$

which by definition of f and r_{μ} can be rewritten to

$$\mathbf{f} \boldsymbol{c} = A \boldsymbol{r} + \begin{pmatrix} 0 \\ \mathbf{f} \boldsymbol{r} \end{pmatrix}.$$

So the opening information we obtain is $x, f, r_z - r'_z$. Note that indeed, $||r||_{\infty} \leq \gamma N m \beta \gamma_{\mathsf{D}}$ and $\deg(r) < 2d_R$ as required.

Lemma 7 (Honest-verifier zero-knowledge). Executions of Π_{OPEN} with an honest verifier can be simulated with statistically indistinguishable distribution.

Proof. The probability that a single coefficient in the prover's randomness causes an abort is p, as defined in Equation (1), p clearly does not depend on any of the prover's secrets. The prover aborts if any of the coefficients causes an abort, so the overall abort probability is $p_{abort} = 1 - (1 - p)^T$ where $T = 2d_R m$ is the number of coefficients used in \mathbf{r}_z .

Therefore, on input c, the simulator first decides to simulate an aborting conversation with probability p_{abort} . In this case, the simulator just outputs $C_{aux}(t)$ for an arbitrary value t of the same length as a basic commitment.

Otherwise, to simulate an accepting conversation, draw a random d from D and a random \mathbf{r}_z subject to $||\mathbf{r}_z||_{\infty} \leq (\gamma Nm)/2 \cdot \beta \gamma_{\mathsf{D}}$, $deg(\mathbf{r}_z) < 2d_R$. Finally, set $\mathbf{t} = C(z; z_r) - d\mathbf{c}$, and commit to \mathbf{t} using the auxiliary commitment scheme. As for correctness of output distribution, note that aborting and non-aborting conversations occur with the correct probabilities. The aborting conversations have statistically indistinguishable distribution by hiding of the auxiliary scheme. The non-aborting ones have exactly the right distribution since the last two messages are directly chosen with the correct distribution and the first follows from the last two. To this end, note that since $d\mathbf{r}$ has degree less than $2d_R$ and norm at most $\beta \gamma_{\mathsf{D}}$, the choice of $\boldsymbol{\rho}$ completely randomizes $\mathbf{r}_z = \boldsymbol{\rho} + d\mathbf{r}$ in the non-aborting case. This follows exactly as in [Lyu09].

4.2 Proof for Opening to a Specific Message.

The protocol Π_{OPEN} demonstrates that the prover knows how to open a commitment to some message, without revealing the randomness. An easy variant, which we will call $\Pi_{\text{OPEN-X}}$, can be used to show that the prover can open \boldsymbol{c} to a specific message x: First we observe that is enough to show that a commitment can be opened to 0, since one can use that protocol on input $\boldsymbol{c} - (0, x)$.

Now, to prove that a commitment can be opened to 0, execute Π_{OPEN} , with the following change: μ is always set to 0. As a result, z = 0 and the verifier checks that this is indeed the case. It trivial to show completeness, special soundness and honest verifier zero-knowledge for this protocol, and we leave this to the reader.

4.3 Proof of Linear Relation

Suppose that the prover has published two commitments $c_1 = C(x_1; r_1), c_2 = C(x_2; r_2)$ and claims that $x_2 = g(x_1)$ for a linear function g. Then consider the following protocol for proving that two commitments contain linearly related vectors:

Protocol Π_{Lin}

- 1. The prover computes commitments $\mathbf{t}_1 = C(\mu_1; \boldsymbol{\rho}_1), \mathbf{t}_2 = C(\mu_2; \boldsymbol{\rho}_2)$ where $\mu_1, \boldsymbol{\rho}_1, \boldsymbol{\rho}_2$ are chosen uniformly from R_q and R^m , resp., subject to $||\boldsymbol{\rho}_1||_{\infty}$, $||\boldsymbol{\rho}_2||_{\infty} \leq (1 + (\gamma Nm)/2)\beta\gamma_{\mathsf{D}}$ and $\deg(\boldsymbol{\rho}_1), \deg(\boldsymbol{\rho}_2) < 2d_R$. Furthermore, set $\mu_2 = g(\mu_1)$. The prover then sends commitments $\mathbf{c}_{aux,1} = C_{aux}(\mathbf{t}_1), \mathbf{c}_{aux,2} = C_{aux}(\mathbf{t}_2)$ to the verifier.
- 2. The verifier sends a random challenge $d \in D$.
- 3. The prover first checks that d is a valid challenge. The prover's goal is to open $t_1 + dc_1$ to $z_1 = \mu_1 + dx_1$ and $r_{z_1} = \rho_1 + dr_1$, and $t_2 + dc_2$ to $z_2 = \mu_2 + dx_2$ and $r_{z_2} = \rho_2 + dr_2$. The protocol is aborted if $||r_{z_1}||_{\infty} > (\gamma Nm/2)\beta\gamma_{\text{D}}$ or $||r_{z_2}||_{\infty} > (\gamma Nm/2)\beta\gamma_{\text{D}}$. Otherwise, the prover sends to the verifier z_1 , r_{z_1}, z_2, r_{z_2} , and opening information $u_{aux,1}$ and $u_{aux,2}$ for the commitments $c_{aux,1}$ and $c_{aux,2}$.
- 4. The verifier checks that $u_{aux,1}$, $u_{aux,2}$ are valid, that $C(z_1; \boldsymbol{r}_{z_1}) = \boldsymbol{t}_1 + d\boldsymbol{c}_1$ and $C(z_2; \boldsymbol{r}_{z_2}) = \boldsymbol{t}_2 + d\boldsymbol{c}_2$, that $g(z_1) = z_2$, that $||\boldsymbol{r}_{z_1}||_{\infty}, ||\boldsymbol{r}_{z_2}||_{\infty} \leq (\gamma Nm/2)\beta\gamma_{\mathsf{D}}$, and that $\deg(\boldsymbol{r}_{z_1}), \deg(\boldsymbol{r}_{z_2}) < 2d_R$.

We now look at the properties of the protocol. The proof is in Appendix A.

Lemma 8. The protocol Π_{LIN} has the following properties:

- Completeness: The verifier always accepts an honest prover when the protocol does not abort. The probability of abort is at most 4/γ.
- Special Soundness: On input two commitments \mathbf{c}_1 , \mathbf{c}_2 and a pair of transcripts $((\mathbf{c}_{aux,1}, \mathbf{c}_{aux,2}), \mathsf{d}, (u_{aux,1}, u_{aux,2}, \mathbf{t}_1, \mathbf{t}_2, z_1, z_2, \mathbf{r}_{z_1}, \mathbf{r}_{z_2})),$ $((\mathbf{c}_{aux,1}, \mathbf{c}_{aux,2}), \mathsf{d}', (u'_{aux,1}, u'_{aux,2}, \mathbf{t}'_1, \mathbf{t}'_2, z'_1, z'_2, \mathbf{r}'_{z_1}, \mathbf{r}'_{z_2}))$ where $\mathsf{d} \neq \mathsf{d}'$, we can extract either a witness for breaking the auxiliary commitment scheme, or valid openings of \mathbf{c}_1 and \mathbf{c}_2 .

- Honest-verifier zero-knowledge: Executions of protocol Π_{LIN} with an honest verifier can be simulated with statistically indistinguishable distribution.

4.4 Proof of Sum

Suppose that the prover has published three commitments $c_1 = C(x_1; r_1)$, $c_2 = C(x_2; r_2)$, $c_3 = C(x_3; r_3)$ and claims that $x_3 = \alpha_1 x_1 + \alpha_2 x_2$ where $\alpha_1, \alpha_2 \in R_q$ are public constants.

Protocol Π_{Sum}

- 1. The prover draws uniform μ_1 , μ_2 from R_q and ρ_i $(i \in \{1, 2, 3\})$ from R^m subject to $||\rho_i||_{\infty} \leq (1 + (\gamma Nm)/2)\beta\gamma_{\mathsf{D}}$ and $\deg(\rho_i) < 2d_R$, and sets $\mu_3 = \alpha_1\mu_1 + \alpha_2\mu_2$. He then computes $\mathbf{t}_i = C(\mu_i; \rho_i)$ and $\mathbf{c}_{aux,i} = C_{aux}(\mathbf{t}_i)$ $(i \in \{1, 2, 3\})$. Finally, the prover sends $\mathbf{c}_{aux,i}$ to the verifier.
- 2. The verifier sends a random challenge $d \in D$.
- 3. The prover first checks that d is a valid challenge. The prover's goal is then to open $t_i + dc_i$ to $z_i = \mu_i + dx_i$ and $r_{z_i} = \rho_i + dr_i$. The protocol is aborted if $||r_{z_i}||_{\infty} > (\gamma Nm/2)\beta\gamma_{\text{D}}$. Otherwise, the prover sends to the verifier z_i , r_{z_i} , and opening information $u_{aux,i}$ for the commitments $c_{aux,i}$.
- 4. The verifier checks that $u_{aux,i}$ are valid, that $C(z_i; \boldsymbol{r}_{z_i}) = \boldsymbol{t}_i + d\boldsymbol{c}_i$, that $z_3 = \alpha_1 z_1 + \alpha_2 z_2$, that $||\boldsymbol{r}_{z_i}||_{\infty} \leq (\gamma Nm/2)\beta\gamma_{\mathsf{D}}$, and that $\deg(\boldsymbol{\rho}_i) < 2d_R$ for $i \in \{1, 2, 3\}$.

Lemma 9. The protocol Π_{SUM} has the following properties:

- Correctness: The verifier always accepts an honest prover when the protocol does not abort. The probability of abort is at most 6/γ.
- Special soundness: On input α_1 , α_2 , three commitments c_1 , c_2 , c_3 and a pair of transcripts $((c_{aux,i})_{i \in \{1,2,3\}}, \mathsf{d}, (u_{aux,i}, t_i, z_i, r_{z_i})_{i \in \{1,2,3\}})$,
- $((c_{aux,i})_{i \in \{1,2,3\}}, \mathsf{d}', (u'_{aux,i}, t'_i, z'_i, r'_{z_i})_{i \in \{1,2,3\}})$ where $\mathsf{d} \neq \mathsf{d}'$, we can extract either a witness for breaking the auxiliary commitment scheme, or valid openings of c_1, c_2, c_3 .
- Honest-verifier zero-knowledge: Executions of protocol Π_{SUM} with an honest verifier can be simulated with statistically indistinguishable distribution.

Proof. An honest prover can clearly answer correctly for any challenge d and hence an honest verifier will always accept if the protocol did not abort. Since each coefficient of $d\mathbf{r}_i$ has norm at most $\beta\gamma_{\mathsf{D}}$, the probability that a single coefficient of \mathbf{r}_{z_i} will cause an abort is

$$\frac{2\beta\gamma_{\mathsf{D}}}{2(1+\gamma Nm/2)\beta\gamma_{\mathsf{D}}+1} \le \frac{2}{\gamma Nm}.$$

Hence the probability that some coefficient of one of the r_{z_i} causes an abort is at most $3 \cdot 2/\gamma$ by the union bound.

The proof of special soundness is similar to that of Lemma 6. If we cannot break the auxiliary commitment scheme, then by the same argument, we can assume that $\mathbf{t}_i = \mathbf{t}'_i$. In this case, one can compute the messages contained in \mathbf{c}_i as $x_i = f^{-1}(z_i - z'_i)$, where f = d - d' and f is again invertible. Then set the randomness $\mathbf{r}_i = \mathbf{r}_{z_i} - \mathbf{r}'_{z_i}$.

For honest-verifier zero-knowledge, first note that the probability p_{abort} that an abort occurs in the protocol is independent of the prover's secret (cf. Lemma 7). But it is well-defined from the parameters β , γ , N and m. Therefore, on input c_i , the simulator first decides to simulate an aborting conversation with probability p_{abort} . In this case, the simulator just outputs $C_{aux}(t_i)$ for arbitrary values t_i of the same length as a basic commitment.

Otherwise, to simulate an accepting conversation, draw a random d from D and random $z_1, z_2, \mathbf{r}_{z_i}$ subject to $||\mathbf{r}_{z_i}||_{\infty} \leq (\gamma Nm)/2 \cdot \beta \gamma_{\text{D}}$. Set $z_3 = \alpha_1 z_1 + \alpha_2 z_2$. Finally, set $\mathbf{t}_i = C(z_i; \mathbf{r}_{z_i}) - d\mathbf{c}_i$, and commit to \mathbf{t}_i using the auxiliary commitment scheme. As for correctness of output distribution, note that aborting and non-aborting conversations occur with the correct probabilities. The aborting conversations have statistically indistinguishable distribution by hiding of the auxiliary scheme. As argued in the proof of Lemma 7, the non-aborting ones have exactly the right distribution since the last two messages are directly chosen with the correct distribution and the first follows from the last two.

4.5 Proving Bounds

Suppose that the prover has published a commitment $\mathbf{c} = C(x; \mathbf{r}_x)$ and claims that the norm of x is small. The idea is to add a short random value μ to x and check whether the sum is sufficiently short. Since the challenge has to be taken into account as well, we can only allow for small challenges, i.e. we restrict the challenge space here to $\{0, 1\}$. This of course increases the soundness error. The protocol can be made efficient in an amortized sense using recent work of Baum et al. [BDLN16] and Cramer and Damgård [CD16].

Let β_x be an upper bound on the norms of all possible x and β_r an upper bound on the norm of the possible μ , where $\beta_r \geq \gamma_x N \beta_x$ for $\gamma_x > 0$.

Protocol Π_{Bound}

- 1. The prover computes a commitment $\boldsymbol{t} = C(\mu; \boldsymbol{\rho})$ for uniform $\mu \in R_q$ and $\boldsymbol{\rho} \in R^m$ subject to $||\mu||_{\infty} \leq \beta_x(1 + \gamma_x N/2), ||\boldsymbol{\rho}||_{\infty} \leq (1 \gamma Nm/2)\beta$ and $\deg(\boldsymbol{\rho}) < 2d_R$, and sends $c_{aux} = C_{aux}(\boldsymbol{t})$ to the verifier.
- 2. The verifier sends a random challenge bit $d \in \{0, 1\}$.
- 3. The prover first checks that d is a valid challenge. The prover's goal is then to open $\mathbf{t} + \mathbf{dc}$ to $z = \mu + dx$ and $\mathbf{z}_r = \boldsymbol{\rho} + \mathbf{dr}$. The protocol is aborted if $||z||_{\infty} > \gamma_x N \beta_x/2$ or $||\mathbf{r}_z||_{\infty} > (\gamma N m/2)\beta$. Otherwise, the prover sends to the verifier z, \mathbf{r}_z , and opening information u_{aux} for the commitment \mathbf{c} .
- 4. The verifier checks that u_{aux} is valid, that $C(z; \mathbf{r}_z) = \mathbf{t} + \mathbf{dc}$, that $||z||_{\infty} \leq (\gamma_x N \beta_x/2)$, that $||\mathbf{r}_z||_{\infty} \leq (\gamma N m/2)\beta$, and that $\deg(\boldsymbol{\rho}) < 2d_R$.

We now look at the properties of Π_{BOUND} . The proof is in Appendix A.

Lemma 10. The protocol Π_{BOUND} has the following properties:

- Correctness: The verifier always accepts an honest prover when the protocol does not abort. The probability of abort is at most $2/\gamma + 2/\gamma_x$.
- Special soundness: On input commitment \mathbf{c} and a pair of transcripts $(\mathbf{c}_{aux}, \mathsf{d}, (u_{aux}, \mathbf{t}, z, \mathbf{r}_z)), (\mathbf{c}_{aux}, \mathsf{d}', (u'_{aux}, \mathbf{t}', z', \mathbf{r}'_z))$ where $\mathsf{d} \neq \mathsf{d}'$, we can extract either a witness for breaking the auxiliary commitment scheme, or a valid opening of \mathbf{c} where the message x has norm at most $\gamma_x N \beta_x$.
- Honest-verifier zero-knowledge: Executions of protocol Π_{BOUND} with an honest verifier can be simulated with statistically indistinguishable distribution.

4.6 Achieving Zero-Knowledge for Dishonest Verifiers

One easy way to have our protocols be zero-knowledge against dishonest verifiers is if a trusted source of random bits is available (which can be implemented via a coin-flipping protocol). One gets the challenge from this source and then clearly honest-verifier zero-knowledge is sufficient.

A different approach is possible if a trapdoor commitment scheme C_{trap} is available, where commitments in this scheme can be equivocated if the trapdoor is known. Then we can transform each of our protocols to a new one that is zero-knowledge: The prover commits to the first message a using C_{trap} , gets the challenge d and then opens $C_{trap}(a)$ and answers d. If the simulator knows the trapdoor, it can make a fake commitment first. Once d arrives, it runs the simulation and equivocates the initial commitment to the value of a that it wants.

5 Instantiations of the Commitment Scheme and Commitment-Friendly Rings

In this section, we make some suggestions for a concrete construction of commitmentfriendly rings and for parameter choices. We recall the definition:

Definition 2. The ring $R_q = R/qR$ is $(\mathsf{D}, \gamma_{\mathsf{D}}, d_R)$ -commitment friendly if there is a subset $\mathsf{D} \subseteq R$ such that for any $\mathsf{c} \in \mathsf{D}$ and $r \in R$ with $\deg(r) < 2d_R$, we have $||\mathsf{c} \cdot r||_{\infty} \leq \gamma_{\mathsf{D}}||r||_{\infty}$. Furthermore, for any $\mathsf{d}, \mathsf{d}' \in \mathsf{D}$ it holds that $\mathsf{d} - \mathsf{d}'$ is invertible modulo q, and $\deg(\mathsf{d}) < d_R$.

In addition, to complete the construction we need to choose a bound β on the randomness for commitments and specify a distribution \mathcal{D} such that it outputs elements in R with norm at most β . Furthermore, \mathcal{D} must be such that the class of functions $\{f_a \mid a \in R_q^m\}$ where $f_a(r) = a \cdot r$ is universal when mapping the support of \mathcal{D} to R_q .

First, we will set $\beta > 1$ to be constant. Our strategy (which is inspired by [BKLP15]) is then to choose the polynomial f(x) in $R_q = \mathbb{F}_q[x]/(f(x))$ in an appropriate way. Note that if f(x) splits in a constant number of distinct irreducible polynomials $f(x) = f_1(x) \cdots f_u(x)$, then R_q is the direct product of u fields. Suppose furthermore that all the f_i have the same degree, then any polynomial of degree less than N/u will be unchanged when reduced modulo any of the f_i , and it will therefore be invertible. This means that a natural candidate for the set D is a set of "short" polynomials of degree less than N/u. For reasons that will become clear later, we will want the degree to be always less than N/3. So we will set $d_R = \min(N/3, N/u)$, and our final definition of D is

$$\mathsf{D} = \{ r \in R \mid ||r||_{\infty} \le \beta, \deg(r) < d_R \}$$

This means that $|\mathsf{D}| \geq \beta^{N/u}$ and since u is constant, our soundness error $1/|\mathsf{D}|$ is negligible.

We define the distribution \mathcal{D} for the prover's randomness to be the uniform distribution over D^m . Now, trivially all outputs have norm at most β and degree less than d_R as required. Jumping ahead, we will later choose $\gamma_{\mathsf{D}} = \beta N$, so that indeed elements in D will have norm less than γ_{D} , as required in Definition 2.

We then consider the functions of form $f_{\boldsymbol{a}}(\boldsymbol{r}) = \boldsymbol{a} \cdot \boldsymbol{r}$ from D^m to R_q and we want to show that they are universal hash functions. Because of the direct product structure of R_q , we can think of the function as the direct product of u functions defined over the fields $\mathbb{F}_i = \mathbb{F}_q[x]/(f_i(x))$. Each of these functions are universal since they compute the dot product over fields and so they have collision probability $1/|\mathbb{F}_i|$. Now since f_a is linear, a collision occurs if and only if the function sends a non-zero input to 0. However, a non-zero vector in $\boldsymbol{r} \in \mathsf{D}^m$ is also non-zero when reduced modulo any $f_i(x)$ because only low degree polynomials occur in \boldsymbol{r} . Hence a collision only occurs if one occurs in each subfield, and so the collision probability is $\prod_i 1/|\mathbb{F}_i| = 1/|R_q|$ – which shows that f_a is universal.

We then need to estimate how much the norm of an element $\mathbf{r} \in R$ is increased when multiplying it by an element $\mathbf{c} \in \mathbf{D}$, i.e. we need the value of the parameter $\gamma_{\mathbf{D}}$. The requirements in Definition 2 ensure that when we compute $\mathbf{d} \cdot \mathbf{r}$, no reductions modulo f(x) take place and so the only increase in norm comes from the polynomial product. It is now easy to see that the increase in norm is by at most a factor $\gamma_{\mathbf{D}} = \beta N$.

In Lemma 1, we saw that breaking binding would produce an Ring-SIS solution of norm at most $2Nm\gamma\beta\gamma_{\mathsf{D}}^2 = 2Nm\gamma\beta\beta^2N^2 \in O(N^3m)$. In the specification of the commitment scheme we required that $\gamma Nm\beta\gamma_{\mathsf{D}}^2 < q$, which is necessary since otherwise the SIS solution we find is certainly not "short". So this means we need $q \in \Omega(N^3m)$.

We now consider the choice of m. For statistical hiding we need that the entropy of the honest committer's randomness is at least $2N \log q + \kappa$ where κ is the statistical security parameter. By definition of the distribution \mathcal{D} this means we need $mN/3 \cdot \log(\beta) = 2N \log q + \kappa$. Since κ is unrelated to the Ring-SIS problem and so does not need to grow with the other parameters, we can safely assume that $N \geq \kappa$ as far as the asymptotics are concerned. So we see that $m \in O(\log q)$ will be sufficient.

We can now summarize what we did in the the following theorem:

Theorem 1. Assume that $R_q = \mathbb{F}_q[x]/(f(x))$, where $\deg(f) = N$ and $f = \prod_{i=1}^{u} f_i(x)$, where u is constant and the $f_i(x)$'s are irreducible (over \mathbb{F}_q), fur-

thermore q is a prime where $q \in \Omega(N^3 \log q)$. Let $d_R = \min(N/3, N/u)$ and $\mathsf{D} = \{r \in R \mid ||r||_{\infty} \leq \beta, \ deg(r) < d_R\}$. Furthermore, in the commitment scheme, let $m \in O(\log q)$, and let $\beta, \gamma > 1$ be constants. If the commitment scheme is set up in this way, it will be statistically hiding, and breaking the binding property implies one can solve the Ring-SIS problem over R_q in time polynomial in N and $\log q$ given a vector of length m as input, and where solutions will have norm in $O(N^3 \log q)$.

It is not a priori clear whether Ring-SIS is going to be hard in all the cases covered by the above theorem. But in some specific cases we can say more. For instance, we may consider the special case that was also used in [BKLP15], where $q \mod 8 = 3$, N is a 2-power, and $f(x) = x^N + 1$. In this case, it is known that f(x) splits in 2 irreducible factors of degree N/2, so this is a special case of our framework. The Ring-SIS problem over rings R_q using this polynomial were studied in [LM06], and under certain conditions on q, m and N it was shown that the Ring-SIS problem in this case is as hard as solving certain problems in ideal lattices in the worst case. These conditions include, of course, that qmust be sufficiently larger than the required norm of the solution for the SIS problem (intuitively, otherwise the solution cannot be claimed to be "short"). If we choose q in $O(N^2 \cdot N^3 \log q) = O(N^5 \log q)$, the conditions are satisfied by our parameter choices, so there is good reason to believe that Ring-SIS is indeed hard for this choice of f(x).

Other choices for f(x) that will also work based on the results in [LM06] include cyclotomic polynomials where we choose q such that f(x) splits in only a small number of irreducible factors. (Note that the factors are all of the same degree.)

Beyond this, one may also consider the much more general case of number rings or orders, for which worst case hardness theorems are also shown in [LM06] and [PR07]. We will work out this option in detail in the final version of this paper.

6 Distributed Key Generation and Decryption

In this section, we describe how to efficiently compile passively secure distributed key generation and decryption protocols into actively secure counterparts for nparties $\mathcal{P} = \{P_1, \ldots, P_n\}$, out of which at most n - 1 can be malicious. Our transformation is generic and applies to schemes that are based on the (Ring-)LWE assumption. The presented protocols are only secure under sequential composition due to the use of rewinding. In Appendix B we will show how to achieve UC security for these protocols.

6.1 Distributed Cryptosystems

We start with an abstract definition of the class of cryptosystems to which our solution applies: let $d, \ell_c, \ell_{\mathsf{pk}}, \ell_{\mathsf{sk}}, \ell_d, \beta_s, \omega_s, \beta_d, \omega_d \in \mathbb{N}, \beta_s, \beta_d \ll q$ (we assume

Parameter	Explanation
P_i	Party i
\mathcal{P}	Set of parties
I	Set of corrupted parties
n	Number of parties
\mathcal{A}	Adversary
d	Dimension of the plaintext space $\{0,1\}^d$ of the cryptosystem
ℓ_s	Length of the randomness that goes into key generation
ℓ_c	Length of a ciphertext
ℓ_{pk}, ℓ_{sk}	Length of the public and private key
ℓ_d	Length of the decryption, before being decoded into a plaintext
β_s, β_d	Maximal norm of randomness used in key generation
	and the noise used in distributed decryption
ω_s, ω_d	"Slack" in norm between honestly chosen vectors and guarantees
	of Π_{Bound} in key generation and decryption
\mathcal{U}^ℓ_eta	Random distribution for vectors of length ℓ and norm at most β
$oldsymbol{F}_a^{ extsf{KG}},oldsymbol{F}_c$	Matrix applied in key generation & decryption

Fig. 2. Parameters used in this Section.

that these parameters implicitly are functions of the computational security parameter λ). An overview over the parameters and notation used in this section can be found in Fig. 2. The parameters of the commitments and zero-knowledge proofs can be found in Fig. 1.

We make the simplifying assumption that ℓ_c , ℓ_{pk} , ℓ_s , ℓ_s , ℓ_d are multiples of the parameter N that was introduced in the definition of the commitment scheme. Let $\mathcal{U}^{\ell}_{\beta}$ be an algorithm that efficiently samples from \mathbb{Z}^{ℓ}_q by choosing each coordinate uniformly at random from $[-\beta,\beta]$ (when representing each \mathbb{Z}_q -element by its representative from (-q/2,q/2]).

The probabilistic encryption algorithm **Enc** maps a string $m \in \{0,1\}^d$ to an element $c \in \mathbb{Z}_q^{\ell_c}$. Moreover, we define generic algorithms KG, Dec for key generation and decryption. These depend on matrices

$$oldsymbol{F}_a^{\mathsf{KG}} \in \mathbb{Z}_q^{(\ell_{\mathsf{pk}}+\ell_{\mathsf{sk}}) imes\ell_s}, oldsymbol{F}_c \in \mathbb{Z}_q^{\ell_d imes\ell_c}$$

and we assume that they are implicitly defined, respectively, by some CRS and the ciphertext c to be decrypted (we may also just sample $\boldsymbol{F}_a^{\mathsf{KG}}$ using a distributed coin-flipping protocol). The decryption additionally uses a publicly known algorithm

$$\mathsf{decode}: \mathbb{Z}_q^{\ell_d} \to \{0,1\}^d \cup \{\bot\}$$

that removes the noise in the ciphertext and differs depending on whether the message is stored in the higher or lower bits of the ciphertext. We then define the key generation and decryption abstractly as being "mostly linear", i.e. both operations consist of multiplying a secret vector with a known public matrix, plus eventually adding some noise. The algorithms KG and Dec are defined as follows:

 $\begin{array}{l} \mathsf{KG}(1^{\lambda}, n, \boldsymbol{F}_{a}^{\mathsf{KG}}, \boldsymbol{s}_{1}, \ldots, \boldsymbol{s}_{n}) \texttt{:} \\ 1. \ \mathrm{For} \ i \in [n] \ \mathrm{compute} \ (\mathsf{pk}_{i}, \mathsf{sk}_{i}) = \boldsymbol{F}_{a}^{\mathsf{KG}} \boldsymbol{s}_{i}. \\ 2. \ \mathrm{Output} \ (\mathsf{pk} = \sum_{i} \mathsf{pk}_{i}, \mathsf{pk}_{1}, \ldots, \mathsf{pk}_{n}, \mathsf{sk}_{1}, \ldots, \mathsf{sk}_{n}). \\ \mathsf{Dec}(\boldsymbol{F}_{c}, \mathsf{sk}_{1}, \ldots, \mathsf{sk}_{n}, \boldsymbol{e}_{1}, \ldots, \boldsymbol{e}_{n}) \texttt{:} \\ 1. \ \mathrm{For} \ i \in [n] \ \mathrm{compute} \ \boldsymbol{d}_{i} = \boldsymbol{e}_{i} + \boldsymbol{F}_{c} \mathsf{sk}_{i}. \\ 2. \ \mathrm{Output} \ (\mathsf{decode}(\sum_{i} \boldsymbol{d}_{i}), \boldsymbol{d}_{1}, \ldots, \boldsymbol{d}_{n}). \end{array}$

We can now define a distributed cryptosystem:

Definition 3 (Distributed Cryptosystem). The tuple of (probabilistic) polynomial-time algorithms D = (KG, Enc, Dec) is a distributed cryptosystem if there exist protocols Π_{KG}, Π_{DEC} that securely implement \mathcal{F}_{KGD} .

Our above definition captures the encryption schemes [BV11,BGV12] directly, but can also be adapted to [FV12] with minor changes in the Dec procedure. Unfortunately it does not directly apply to [HPS98] due to the structure of pk. We refer to [CS16] for an overview over the mentioned schemes.

For those schemes mentioned above, \mathcal{F}_{KGD} can easily be implemented with security against passive adversaries. In the case of active adversaries, we have to ensure that s_i , e_i are bounded as in \mathcal{F}_{KGD} . Moreover, pk_i , sk_i , d_i of the dishonest parties may depend on those values of the honest parties, and they may not be computed using \mathbf{F}_a^{KG} , \mathbf{F}_c at all. We remark that the parameters ω_s , ω_d allow the adversary in the dishonest case to choose slightly larger values than in the honest case. This is necessary because Π_{BOUND} naturally comes with some tightness slack.

6.2 Actively Secure Key Generation

The key generation protocol can informally be described as follows: in a first step, we let all parties sample a value s_i that they commit to. They then prove in zero-knowledge that this commitment indeed contains a short value. We moreover let each party commit to the values pk_i , sk_i as they can be computed from s_i and let them prove that the commitments can indeed be obtained using the public linear transform F_a^{KG} . As was shown in the previous section, all of these steps can be done efficiently. Finally we let the parties open pk_i , so that they then individually can compute the public key locally.

To ease notation, we can rewrite the above definition as

$$oldsymbol{F}_a^{\mathsf{KG}} = egin{pmatrix} oldsymbol{F}_a^{\mathsf{pk}} \ oldsymbol{F}_a^{\mathsf{sk}} \end{pmatrix},$$

where $\boldsymbol{F}_{a}^{\mathsf{pk}} \in \mathbb{Z}_{q}^{\ell_{\mathsf{pk}} \times \ell_{s}}, \boldsymbol{F}_{a}^{\mathsf{sk}} \in \mathbb{Z}_{q}^{\ell_{\mathsf{sk}} \times \ell_{s}}$. Since we made the simplifying assumption that all matrix dimensions are multiples of N, we can decompose $\boldsymbol{F}_{a}^{\mathsf{pk}}, \boldsymbol{F}_{a}^{\mathsf{sk}}$ into

Functionality \mathcal{F}_{KGD}

Key Generation:

- 1. Wait for each party P_i to input (KeyGen, F_a^{KG}).
- 2. For each $P_i \in \mathcal{I}$, \mathcal{A} inputs s_i . If $s_i \notin \mathbb{Z}_q^{\ell_s}$ or $||s_i||_{\infty} > \omega_s \cdot \beta_s$, then output (Abort, P_i) to all honest parties and stop.
- 3. For each $P_i \in \mathcal{P} \setminus \mathcal{I}$ sample $\boldsymbol{s}_i \stackrel{s}{\leftarrow} \mathcal{U}_{\beta_s}^{\ell_s}$.
- 4. Compute $(\mathsf{pk}, \mathsf{pk}_1, \dots, \mathsf{pk}_n, \mathsf{sk}_1, \dots, \overset{\scriptscriptstyle Fs}{\mathsf{sk}}_n) \leftarrow \mathsf{KG}(1^{\lambda}, n, F_a^{\mathsf{KG}}, s_1, \dots, s_n).$
- 5. Locally store $(Keys, pk, sk_1, ..., sk_n)$ if no such pk has been stored before.
- 6. Output $(\mathsf{pk}, (\mathsf{pk}_i)_{P_i \in \mathcal{P} \setminus \mathcal{I}})$ to \mathcal{A} and $(\mathsf{pk}, \mathsf{sk}_i)$ to each honest P_i .

Decryption:

- 1. Wait for each party P_i to input (Decrypt, pk, F_c).
- 2. Load (Keys, pk, sk_1 , ..., sk_n). If no such entry can be found, abort.
- 3. For each $P_i \in \mathcal{I} \ \mathcal{A}$ inputs e_i . If $e_i \notin \mathbb{Z}_q^{\ell_d}$ or $||e_i||_{\infty} > \omega_d \cdot \beta_d$ then output (Abort, P_i) to all honest parties and stop.
- 4. For each $i \in \mathcal{P} \setminus \mathcal{I}$ sample $\boldsymbol{e}_i \stackrel{\$}{\leftarrow} \mathcal{U}_{\beta_d}^{\ell_d}$.
- 5. Compute $(m, d_1, \ldots, d_n) \leftarrow \mathsf{Dec}(\check{F}_c^a, \mathsf{sk}_1, \ldots, \mathsf{sk}_n, e_1, \ldots, e_n).$
- 6. Output $(m, (\mathbf{d}_i)_{P_i \in \mathcal{P} \setminus \mathcal{I}})$ to each dishonest P_i and m to each honest P_i .

Fig. 3. \mathcal{F}_{KGD} : Ideal functionality for distributed key generation.

Protocol Π_{KG}

- 1. Each P_i locally samples $s_i \stackrel{\$}{\leftarrow} \mathcal{U}_{\beta_c}^{\ell_s}$.
- 2. Each P_i computes and broadcasts the commitments $C(\mathbf{s}_i), C(\mathbf{F}_a^{\mathsf{pk}} \mathbf{s}_i), C(\mathbf{F}_a^{\mathsf{pk}} \mathbf{s}_i)$.
- 3. Each P_i uses the following proofs towards all parties. Sample the challenge using $\mathcal{F}_{\text{RAND}}$:
 - (a) Π_{BOUND} on $C(\boldsymbol{s}_i)$ to show that $||\boldsymbol{s}_i||_{\infty} \leq \beta_s$.
 - (b) Π_{LIN} on $C(\boldsymbol{s}_i)$, $C(\boldsymbol{F}_a^{\text{pk}}\boldsymbol{s}_i)$ using $\boldsymbol{F}_a^{\text{pk}}$.
 - (c) Π_{LIN} on $C(\boldsymbol{s}_i), C(\boldsymbol{F}_a^{\mathsf{sk}}\boldsymbol{s}_i)$ using $\boldsymbol{F}_a^{\mathsf{sk}}$.
 - If one of the proofs fails then abort.
- 4. Denote with pk_i the committed value in $C(\mathbf{F}_a^{\mathsf{pk}}\mathbf{s}_i)$. Each P_i proves to all parties that $C(\mathbf{F}_a^{\mathsf{pk}}\mathbf{s}_i)$ contains pk_i using $\Pi_{\mathsf{OPEN-X}}$. If one of the proofs fails, then abort.
- 5. If all proofs were correct, then output $\mathsf{pk} = \sum_{i \in [n]} \mathsf{pk}_i$.

Fig. 4. $\Pi_{\rm KG}$: Protocol for actively secure key generation.

submatrices of size $N \times N$. Moreover, let \boldsymbol{r} be a vector $\boldsymbol{r} = (\boldsymbol{r}_1 | \dots | \boldsymbol{r}_k)^{\top}$ where each $\boldsymbol{r}_i \in R_q$, then $C(\boldsymbol{r})$ is an abbreviation for the list of commitments to each individual \boldsymbol{r}_i . This gives an intuitive way to extend $\Pi_{\text{OPEN-X}}$, Π_{SUM} and Π_{BOUND} to longer vectors. We therefore implicitly assume that if we apply Π_{LIN} to a matrix \boldsymbol{F} and $C(\boldsymbol{r})$, then the appropriate number of individual instances of Π_{LIN} with the respective submatrices of \boldsymbol{F} is being used.

Using this above generalization, we can instantiate KG with active security as shown in Fig. 4. We assume that there exists a coin-flipping functionality $\mathcal{F}_{\text{RAND}}$ because the zero-knowledge protocols are only honest-verifier zero-knowledge.

The commitments $C(\mathsf{sk}_i) := C(\mathbf{F}_a^{\mathsf{sk}} \mathbf{s}_i)$ will be saved for later: we will use it again in the distributed decryption.

Protocol Π_{Dec}

The parties in \mathcal{P} want to decrypt the ciphertext c. P_i has sk_i . The commitment $C(\mathsf{sk}_i)$ is known to each $P_j \in \mathcal{P}$.

- 1. Each P_i locally samples $e_i \stackrel{\$}{\leftarrow} \mathcal{U}_{\beta_d}^{\ell_d}$.
- 2. Each P_i derives F_c from c and computes and broadcasts the commitments

$$C(\mathbf{F}_c \mathsf{sk}_i), C(\mathbf{e}_i), C(\mathbf{F}_c \mathsf{sk}_i + \mathbf{e}_i)$$

- 3. Each P_i uses the following proofs towards all parties. Sample the challenge using $\mathcal{F}_{\text{RAND}}$:
 - (a) Prove using Π_{BOUND} about $C(\boldsymbol{e}_i)$ that $||\boldsymbol{e}_i||_{\infty} \leq \beta_d$.
 - (b) Prove using Π_{LIN} that $C(\boldsymbol{F}_c \mathsf{sk}_i)$ is the linear transform of $C(\mathsf{sk}_i)$ when applying \boldsymbol{F}_c .
 - (c) Prove using Π_{SUM} that $C(\mathbf{F}_c \mathsf{sk}_i + \mathbf{e}_i)$ is the sum of $C(\mathbf{F}_c \mathsf{sk}_i)$ and $C(\mathbf{e}_i)$. If one of the proofs fails then abort.
- 4. Each P_i broadcasts d_i and proves towards all parties that $C(F_c \mathsf{sk}_i + e_i)$ opens as d_i using $\Pi_{\text{OPEN-X}}$.
- 5. If all proofs were correct then output $m \leftarrow \mathsf{decode}(\sum_{i \in [n]} d_i)$.

Fig. 5. Π_{Dec} : Protocol for the actively secure decryption of a ciphertext.

6.3 Actively Secure Distributed Decryption

An actively secure version of Dec can be obtained using a similar compilation step to the one that turned KG into $\Pi_{\rm KG}$. The main difference lies in the computed values and in the zero-knowledge proofs that are applied. We moreover use the commitments $C(\mathsf{sk}_i)$ that are publicly known in the process. This allows that each party can prove that it has applied \mathbf{F}_c to the correct key share. The protocol can be found in Fig. 5.

6.4 Security of Π_{KG} and Π_{Dec}

We now show that the above two protocols can be used to implement \mathcal{F}_{KGD} with active security.

Theorem 2. The protocols Π_{KG} , Π_{DEC} implement \mathcal{F}_{KGD} in the standalone setting with security against static active adversaries corrupting up to n-1 parties in the $\mathcal{F}_{\text{RAND}}$ -hybrid model with auxiliary commitments and broadcast.

A simulator for the protocols is provided in Fig. 6. In the proof, we will argue about the indistinguishability of certain distributions, where $\stackrel{c}{\approx}$ symbolizes that

Simulator S_{KGD}

Key Generation:

- 1. Wait for $\boldsymbol{\mathcal{A}}$ to input the set $\boldsymbol{\mathcal{I}}$ of corrupted parties.
- 2. For each honest $P_i \in \mathcal{P} \setminus \mathcal{I}$ choose $\boldsymbol{s}_i \leftarrow \mathcal{U}_{\beta_s}^{\ell_s}$.
- 3. For each honest P_i compute the commitments $C(\mathbf{s}_i), C(\mathbf{F}_a^{\mathsf{pk}}\mathbf{s}_i), C(\mathbf{F}_a^{\mathsf{sk}}\mathbf{s}_i)$ and send them to all dishonest parties P_j .
- 4. For each honest party P_i perform the zero-knowledge proofs in Step (3) of Π_{KG} honestly. Abort if the protocol aborts.
- 5. Rewind \mathcal{A} for the proofs of Π_{BOUND} to extract s_j for all dishonest parties. Change the output of $\mathcal{F}_{\text{RAND}}$ to achieve extraction.
- 6. Also rewind \mathcal{A} to extract the witnesses from Π_{Lin} . If they do not match with the extracted s_j then abort.
- 7. Submit all the s_j of the dishonest parties to \mathcal{F}_{KGD} and obtain pk_i .
- 8. During Step (4) open each $C(\mathbf{F}_{a}^{\mathsf{pk}}\mathbf{s}_{i})$ as pk_{i} by simulating $\Pi_{\mathsf{OPEN-x}}$. Therefore fix the challenge in advance using $\mathcal{F}_{\mathsf{RAND}}$.
- 9. For all dishonest parties in Step (4) also abort if the extracted witness of $C(\mathbf{F}_a^{\mathsf{pk}}\mathbf{s}_j)$ disagrees with the value pk_j announced by P_j .

Distributed Decryption:

- 1. The set of dishonest parties \mathcal{I} is the same as in Π_{KG} . Let $\mathsf{sk}_i := \mathbf{F}_a^{\mathsf{sk}} \mathbf{s}_i$ and $C(\mathsf{sk}_i) = C(\mathbf{F}_a^{\mathsf{sk}} \mathbf{s}_i)$ be the same commitment as in the instance of Π_{KG} .
- 2. Sample $e_i \stackrel{\$}{\leftarrow} \mathcal{U}^{\ell_d}_{\beta_d}$ for each honest P_i .
- 3. Compute the commitments $C(\mathbf{F}_c \mathbf{s} \mathbf{k}_i)$, $C(\mathbf{e}_i)$, $C(\mathbf{F}_c \mathbf{s} \mathbf{k}_i + \mathbf{e}_i)$ honestly for all honest P_i , then broadcast them.
- 4. Run Step (3) honestly with the correct inputs for the honest parties.
- 5. In Step (3) use rewinding for the dishonest parties to extract the witnesses for Π_{BOUND} , Π_{LIN} , Π_{SUM} . If a witness is not compatible with sk_j then abort. Also abort if the protocol aborts.
- 6. Rename the witnesses of the dishonest P_j from Π_{SUM} as d_j . Send these to \mathcal{F}_{KGD} . Obtain d_i for all honest P_i from \mathcal{F}_{KGD} .
- 7. In Step (4) simulate the opening of $C(\mathbf{F}_c \mathsf{sk}_i + \mathbf{e}_i)$ using $\Pi_{\text{OPEN-X}}$ as \mathbf{d}_i by adjusting the output of $\mathcal{F}_{\text{RAND}}$.
- 8. For all dishonest parties in Step (4) abort if they prove that the value inside $C(\mathbf{F}_c \mathbf{s} \mathbf{k}_j + \mathbf{e}_j)$ is different from \mathbf{d}_j as extracted before.

Fig. 6. S_{KGD} : Simulator for the protocols $\Pi_{\text{KG}}, \Pi_{\text{Dec}}$.

two distributions are *computationally indistinguishable*. Similarly, we use $\stackrel{s}{\approx}, \stackrel{p}{\approx}$ if the distributions are statistically close or perfectly indistinguishable.

Proof. We will first prove security of Π_{KG} by showing that the distribution τ_{Π} of protocol transcripts of Π_{KG} is indistinguishable from the distribution τ_{SIM} of outputs of \mathcal{S}_{KGD} using a sequence of hybrids.

Key generation. We start by defining $\tau_{1,\text{KG}}$ to be the same simulator as τ_{SIM} except that it now aborts if the proofs in Steps (3) – (4) of Π_{KG} are not correct, i.e. we do not specifically check the relations on the extracted data anymore.

Because $\Pi_{\text{LIN}}, \Pi_{\text{OPEN-X}}$ are computationally sound we get that $\tau_{\text{SIM}} \approx \tau_{1,\text{KG}}$. Define $\tau_{2,\text{KG}}$ to be the same as $\tau_{1,\text{KG}}$ except that we now simulate the proofs in Step (3) of Π_{KG} . Because $\Pi_{\text{BOUND}}, \Pi_{\text{LIN}}$ are statistical zero-knowledge, it follows that $\tau_{1,\text{KG}} \approx \tau_{2,\text{KG}}$.

Now we start from the other side: define $\tau'_{1,\text{KG}}$ to be the same as τ_{Π} except that we replace the honest proofs in Steps (3) – (4) with simulations. Therefore $\tau_{\Pi} \stackrel{s}{\approx} \tau'_{1,\text{KG}}$. Because we do now not need witnesses anymore, we can define $\tau'_{2,\text{KG}}$ to be the same as $\tau'_{1,\text{KG}}$ except that the commitments in Step (2) are replaced with those generated by S_{KGD} . By the statistical hiding of the commitment scheme, it holds that $\tau'_{1,\text{KG}} \stackrel{s}{\approx} \tau'_{2,\text{KG}}$. Moreover, the distributions of $\tau_{2,\text{KG}}$ and $\tau'_{2,\text{KG}}$ are identical and the claim follows.

Decryption. We start similarly as in the Π_{KG} case: first, let $\tau_{1,\text{DEC}}$ be a simulator that does the same as \mathcal{S}_{KGD} , but aborts in Steps (3) – (4) only if one of the proofs aborts. We obtain hat $\tau_{\text{SIM}} \stackrel{c}{\approx} \tau_{1,\text{DEC}}$ due to the computational binding property. In particular, this means that in $\tau_{1,\text{DEC}}$ the adversary must use the correct decryption key and succeeds using another one only by breaking the binding property of our scheme. We then define $\tau_{2,\text{DEC}}$ to be the same as $\tau_{1,\text{DEC}}$, just that the proofs in the simulation are now simulated by programming $\mathcal{F}_{\text{RAND}}$ appropriately, which yields $\tau_{1,\text{DEC}} \stackrel{s}{\approx} \tau_{2,\text{DEC}}$. Similarly as above, we define $\tau'_{1,\text{DEC}}$ to be the same as τ_{Π} where we now simulate the zero-knowledge proofs. This implies $\tau'_{1,\text{DEC}} \stackrel{s}{\approx} \tau_{\Pi}$. But observe that we can then again replace the commitments $C(e_i)$, $C(\mathbf{F}_c \mathbf{sk}_i + \mathbf{e}_i)$ generated in Step (2) with those that were used in Step (3) of the Simulator \mathcal{S}_{KGD} . Due to the statistical hiding property of $C(\cdot)$, it follows that $\tau'_{1,\text{DEC}} \stackrel{s}{\approx} \tau'_{2,\text{DEC}}$. We now observe that $\tau'_{2,\text{DEC}} \stackrel{p}{\approx} \tau_{2,\text{DEC}}$ due to their construction, which concludes the proof.

Optimizing away some of the proofs In practice we can, with a careful choice of parameters, avoid using the proof Π_{SUM} : opening the sum $C(a+b, r_1+r_2) = C(a, r_1) + C(b, r_2)$ of two commitments $C(a, r_1), C(b, r_2)$ leaks information about the individual randomness r_1, r_2 , thereby breaking the security. This is why we use Π_{SUM} in the protocols to prove that a commitment opens to the sum of two other commitments.

On the other hand, if we open $C(a+b, r_1+r_2)$ using $\Pi_{\text{OPEN-x}}$ then only a+b is revealed, thereby not leaking information about the randomness of the terms anyway. As an optimization one can therefore avoid the use of Π_{SUM} and simply add commitments directly, as long as the number of terms is small enough such that the randomness does not grow too large (which would break the binding of $C(\cdot)$).

6.5 On the Efficiency of the Protocols

It remains open how to instantiate the above protocols, namely how to choose ω_d, ω_s based on the properties of the zero-knowledge proofs. For the sake of simplicity, we will assume that each matrix or committed vector in Π_{KG} has length

N, i.e. we will invoke each zero-knowledge proof only for one witness. First, we observe from Π_{BOUND} that $\omega_s = N \cdot \gamma_x$. That is, the tightness of Π_{KG} crucially depends upon the correctness of Π_{BOUND} . When not using amortization techniques, we will run $\Pi_{\text{BOUND}} \kappa$ times in parallel to achieve good enough soundness. By a union bound, at least one of these κ instances fails with probability at most $p_{abort} \leq \kappa \cdot (2/\gamma + 2/\gamma_x)$.

Assuming we want p_{abort} to be constant (which means that in the worst case, we may have to repeat the above experiment $O(\kappa)$ times) then this implies that $\gamma, \gamma_x = O(\kappa)$. While this yields a very tight bound of $\omega_s = O(N \cdot \kappa)$, in the worst case we may have to run Π_{BOUND} up to κ^2 times. It can easily be verified that one needs at most $O(\kappa/\log(\kappa))$ instances of Π_{LIN} and $\Pi_{\text{OPEN-X}}$. As a consequence, by lowering the error probability of all the κ parallel zero-knowledge proofs to $1/\text{poly}(\kappa)$ one can reduce the total number of instances of Π_{BOUND} to $O(\kappa^2/\log(\kappa))$ at the expense of setting $\omega_s = N \cdot \text{poly}(\kappa)$.

In the case of Π_{DEC} we could moreover do the following: assume we want to decrypt multiple ciphertexts simultaneously, let's say $O(\log(\kappa))$ many. This then allows to use the amortization technique from [BDLN16], which lowers the number of instances of Π_{BOUND} per instance of Π_{DEC} to $O(\kappa)$. Unfortunately this yields a larger bound on ω_d , namely $\omega_d = O(N^2 \cdot \kappa^{O(\log(\kappa))})$. Using the recent improvement of this technique due to [CD16], one can decrypt $O(\kappa^2)$ simultaneously using Π_{DEC} with $\omega_d = O(N \cdot \kappa)$. Observe that these proofs can actually be performed before decryption needs to be done. This is because they are independent of the decrypted value, hence preprocessing them allows to circumvent the need to run multiple instances of Π_{DEC} at once.

6.6 Threshold Protocols for other Lattice-based Primitives

It might be tempting to hope that the above techniques can also be used to give more efficient protocols for e.g. threshold signatures. There are (currently) two main approaches for lattice signatures, namely Fiat-Shamir style protocols like [Lyu09] or those that use a hash-and-sign approach such as [GPV08]. In the first case, such signature schemes have a rejection-sampling step where a bit is chosen with a certain abort probability that depends both on the signature and the secret basis, which is the signing key. This requires computation with very high precision. For hash-and-sign type constructions, the signer has to sample a short lattice vector using a trapdoor. It has been shown [BKP13] that this can actually be done in a distributed fashion, but the approach requires that all parties sample shares according to a Gaussian distribution. It is an interesting open question how to perform this efficiently with active security. For both cases, we do not see how our commitment scheme could be applied to solve the actual bottlenecks of the threshold versions.

References

AJL⁺12. Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EU-ROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501. Springer, 2012.

- BCK⁺14. Fabrice Benhamouda, Jan Camenisch, Stephan Krenn, Vadim Lyubashevsky, and Gregory Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In Advances in Cryptology - ASIACRYPT 2014, pages 551–572, 2014.
- BDLN16. Carsten Baum, Ivan Damgård, Kasper Larsen, and Michael Nielsen. How to prove knowledge of small secrets. In *Advances in Cryptology-CRYPTO* 2016. Springer, 2016.
- BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 309–325. ACM, 2012.
- BKLP15. Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In *Computer Security - ESORICS 2015*, pages 305– 325, 2015.
- BKP13. Rikke Bendlin, Sara Krehbiel, and Chris Peikert. How to share a lattice trapdoor: Threshold protocols for signatures and (H)IBE. In Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings, pages 218–236, 2013.
- Blu82. Manuel Blum. Coin flipping by telephone A protocol for solving impossible problems. In COMPCON'82, Digest of Papers, Twenty-Fourth IEEE Computer Society International Conference, San Francisco, California, USA, February 22-25, 1982, pages 133–137, 1982.
- BRS02. John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers, pages 62–75, 2002.
- BV11. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In Advances in Cryptology-CRYPTO 2011, pages 505–524. Springer, 2011.
- CD16. Ronald Cramer and Ivan Damgård. Amortized complexity of zero-knowledge proofs revisited: Achieving linear soundness slack. Cryptology ePrint Archive, Report 2016/681, 2016. http://eprint.iacr.org/2016/681.
- CS16. Ana Costache and Nigel P. Smart. Which ring based somewhat homomorphic encryption scheme is best? In Topics in Cryptology CT-RSA 2016
 The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 March 4, 2016, Proceedings, pages 325–340, 2016.
- DF89. Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings, pages 307-315, 1989.
- DKL⁺13. Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P. Smart. Practical covertly secure MPC for dishonest majority or: Breaking the SPDZ limits. In Computer Security ESORICS 2013 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings, pages 1–18, 2013.

- DPP93. Ivan Damgård, Torben P. Pedersen, and Birgit Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. In Advances in Cryptology - CRYPTO '93, pages 250–265, 1993.
- DPSZ12. Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty Computation from Somewhat Homomorphic Encryption. In Proceedings of Crypto, pages 643–662, Springer Verlag 2012.
- FV12. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.
- GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008, pages 197–206, 2008.
- HPS98. Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. Ntru: A ring-based public key cryptosystem. In *Algorithmic number theory*, pages 267–288. Springer, 1998.
- JKPT12. Abhishek Jain, Stephan Krenn, Krzysztof Pietrzak, and Aris Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In Advances in Cryptology - ASIACRYPT 2012, pages 663–680, 2012.
- KTX08. Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings, pages 372– 389, 2008.
- LM06. Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In Automata, Languages and Programming, pages 144–155. Springer, 2006.
- LSSV16. Yehuda Lindell, Nigel P. Smart, and Eduardo Soria-Vazquez. More efficient constant-round multi-party computation from bmr and she. Cryptology ePrint Archive, Report 2016/156, 2016.
- Lyu08. Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In *Public Key Cryptography–PKC 2008*, pages 162–179. Springer, 2008.
- Lyu09. Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings, pages 598-616, 2009.
- PR07. Chris Peikert and Alon Rosen. Lattices that admit logarithmic worst-case to average-case connection factors, pages 478–487. 2007.
- Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In STOC, pages 84–93, 2005.
- XXW13. Xiang Xie, Rui Xue, and Minqian Wang. Zero knowledge proofs from ringlwe. In Cryptology and Network Security - 12th International Conference, CANS 2013, pages 57–73, 2013.

A Zero-Knowledge Proofs (Continued)

A.1 Proof of linear relation

Proof (Lemma 8). An honest prover can clearly answer correctly for any challenge d and hence an honest verifier will always accept if the protocol did not abort. Since each coefficient of $d\mathbf{r}$ has norm at most $\beta\gamma_{\rm D}$, the probability that a single coefficient of \mathbf{r}_{z_2} will cause an abort is

$$\frac{2\beta\gamma_{\mathsf{D}}}{2(1+\gamma Nm/2)\beta\gamma_{\mathsf{D}}+1} \le \frac{2}{\gamma Nm}$$

The probability for coefficients of r_{z_2} is the same. Hence the probability that some coefficient of either r_{z_1} or r_{z_2} causes an abort is at most $4/\gamma$, by the union bound.

The proof of special soundness is similar to that of Lemma 6. If we cannot break the auxiliary commitment scheme, then by the same argument, we can assume that $t_1 = t'_1$ and $t_2 = t'_2$. In this case, one can compute the messages contained in c_1 as $x_1 = f^{-1}z_1 - z'_1$ and in c_2 as $x_2 = f^{-1}z_2 - z'_2$, where f = d - d' and f is again invertible. Then set the randomness $r_1 = r_{z_1} - r'_{z_2}$ and $r_2 = r_{z_2} - r'_{z_2}$.

For honest-verifier zero-knowledge, note that the probability p_{abort} that an abort occurs in the protocol is independent of the prover's secret. But it is well defined from the parameters β , γ , N and m. Therefore, on input c_1 , c_2 , the simulator first decides to simulate an aborting conversation with probability p_{abort} . In this case, the simulator just outputs $C_{aux}(s)$ and $C_{aux}(t)$ for arbitrary values s and t of the same length as a basic commitment.

Otherwise, to simulate an accepting conversation, draw a random d from D and random z_1 , \mathbf{r}_{z_1} , \mathbf{r}_{z_2} subject to $||\mathbf{r}_{z_1}||_{\infty}$, $||\mathbf{r}_{z_2}||_{\infty} \leq (\gamma Nm)/2 \cdot \beta \gamma_{\text{D}}$. Set $z_2 = g(z_1)$. Finally, set $\mathbf{t}_1 = C(z_1; \mathbf{r}_{z_1}) - \mathbf{dc}_1$ and $\mathbf{t}_2 = C(z_2; \mathbf{r}_{z_2}) - \mathbf{dc}_2$, and commit to \mathbf{t}_1 and \mathbf{t}_2 using the auxiliary commitment scheme. As for correctness of output distribution, note that aborting and non-aborting conversations occur with the correct probabilities. The aborting conversations have statistically indistinguishable distribution by hiding of the auxiliary scheme. As argued in the proof of Lemma 7, the non-aborting ones have exactly the right distribution and the first follows from the last two.

A.2 Proving bounds

Proof (Lemma 10). An honest prover can clearly answer correctly for any challenge d and hence an honest verifier will always accept if the protocol did not abort. Note that the challenge d is a bit in this case and hence has norm at most 1. Since each coefficient of an element in R_q has norm at most β , the probability that a single coefficient of \mathbf{r}_z will cause an abort is

$$\frac{2\beta}{2(1+\gamma Nm/2)\beta+1} \le \frac{2}{\gamma Nm}$$

The probability that a single coefficient of z will cause an abort is

$$\frac{2\beta_x}{2(1+\gamma_x N/2)\beta_x + 1} \le \frac{2}{\gamma_x N}$$

since $z \in R_q$ and each coefficient of z has norm at most $\gamma_x N \beta_x$. Hence by the union bound, the probability that some coefficient of z or r_z causes an abort is at most $2/\gamma + 2/\gamma_x$.

The proof of special soundness is similar to that of Lemma 6. If we cannot break the auxiliary commitment scheme, then by the same argument, we can assume that $\boldsymbol{t} = \boldsymbol{t}'$. In this case, one can compute the message contained in \boldsymbol{c} as x = z - z'. Then set the randomness $\boldsymbol{r} = \boldsymbol{r}_z - \boldsymbol{r}'_z$. Note that indeed, $||\boldsymbol{x}||_{\infty} \leq \gamma_x N \beta_x$, $||\boldsymbol{r}||_{\infty} \leq \gamma N m \beta \leq \gamma N m \beta \gamma_{\mathsf{D}}$, and $\deg(\boldsymbol{r}) < 2d_R$ as required.

For honest-verifier zero-knowledge, note that the probability p_{abort} that an abort occurs in the protocol is independent of the prover's secret. But it is well defined from the parameters β , γ , N, m, β_x , and γ_x . Therefore, on input c, the simulator first decides to simulate an aborting conversation with probability p_{abort} . In this case, the simulator just outputs $C_{aux}(t)$ for an arbitrary value t of the same length as a basic commitment.

Otherwise, to simulate an accepting conversation, draw a random d from $\{0,1\}$ as well as random z and r_z subject to $||z||_{\infty} \leq \gamma_x N \beta_x/2$ and $||r_z||_{\infty} \leq (\gamma N m)/2 \cdot \beta$. Finally, set $t = C(z; r_z) - dc$ and commit to t using the auxiliary commitment scheme. As for correctness of output distribution, note that aborting and non-aborting conversations occur with the correct probabilities. The aborting conversations have statistically indistinguishable distribution by hiding of the auxiliary scheme. As argued in the proof of Lemma 7, the non-aborting ones have exactly the right distribution since the last two messages are directly chosen with the correct distribution and the first follows from the last two.

B More Companion Protocols

For all practical purposes, the protocols Π_{KG} , Π_{Dec} from the previous section are not satisfactory. We will now improve them in multiple ways: in a first step, an extension to achieve UC security will be discussed. Moreover, we show a simple approach that allows to compute encryptions of powers of sk. This in turn can be used in an alternative distributed decryption algorithm that uses optimistic decryption. The protocols in this appendix are presented without proofs: their actual security depends on details of the schemes and chosen parameters which would complicate the presentation without yielding any new insights, and the basic structure of the protocols follows those from Section 6.

B.1 Some Further Assumptions

The starting point is to make some further assumptions about D.Enc. In Section 6, we only assumed that such an algorithm exists, while we now require

that the encryption algorithm itself can be modeled in a similar way as KG, Dec - namely, that it can be described in terms of linear operations.

Similarly to the message space R_q of $C(\cdot)$ we define the message space of Enc as R_p for $p \ll q$ (instead of $\{0,1\}^d$). By representing the coefficients of the elements as integers from the interval (-p/2, p/2] we can naturally embed each $m \in R_p$ into \mathbb{Z}_q^N . In particular, for a small enough number of ring operations in R_p we can simulate these operations on the embedding into \mathbb{Z}_q^N , namely, for as long as the coefficients do not get too big and wrap around modulo q.

We assume that $\ell_e, \beta_e \in \mathbb{N}, \beta_e \ll q$ and N divides ℓ_e . Similarly as before, ℓ_e is the length of the randomness vector and β_e is the maximal norm of the randomness used in Enc, that we will now also describe in terms of linear operations: given a public key pk, we require that there is a deterministic algorithm to compute two matrices $F_e^{\mathsf{pk}} \in \mathbb{Z}_q^{\ell_c \times \ell_e}$, $F_e^m \in \mathbb{Z}_q^{\ell_c \times N}$, chosen independently of the plaintext and the noise, such that Enc, on an input $m \in R_p, e \in \mathbb{Z}_q^{\ell_e}$ with $||e||_{\infty} \leq \beta_e$, performs the following operations:

 $Enc(\boldsymbol{F}_{e}^{pk}, \boldsymbol{F}_{e}^{m}, m, \boldsymbol{e})$:

- 1. Consider m as representation in \mathbb{Z}_q^N , which we denote m. 2. Compute $c = \mathbf{F}_e^{\mathbf{pk}} \mathbf{e} + \mathbf{F}_e^m \mathbf{m}$.
- 3. Output c.

In a nutshell, the above allows us to encrypt values we committed to *inside* the commitment without revealing them. A direct consequence of the above representation is that, if we assume the embedding of m to be homomorphic, that $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(m_1) + \mathsf{Enc}_{\mathsf{pk}}(m_2)) = m_1 + m_2$. Depending on the relationship between p, q and β_e , we may allow a number of such additions before Dec yields an incorrect value.

Parameter	Explanation
ℓ_e	Length of randomness vector for encryption
β_e	Noise bound for randomness in encryption
p	Modulus of plaintext space
R_p	Plaintext space of D
$oldsymbol{F}_{e}^{pk}$	Matrix applied to randomness vector in encryption
$oldsymbol{F}_{e}^{m}$	Matrix applied to message in encryption
$oldsymbol{F}_{c}^{m},oldsymbol{f}_{c}^{m}$	Values used to multiply ciphertext c by a constant
β_m	Noise bound for rerandomization of products with a constant
$\overline{pk}_i, \overline{sk}_i$	Public/private key pair of the party P_i

Fig. B.1. Additional Parameters used in this Appendix.

An additional requirement is that, given a ciphertext $c \in \mathbb{Z}_q^{\ell_c}$ there exists a deterministic algorithm to compute $\mathbf{F}_c^m \in \mathbb{Z}_q^{\ell_c \times N}$, $\mathbf{f}_c^m \in \mathbb{Z}_q^{\ell_c}$ from c and independently of a such that $\mathsf{Dec}_{\mathsf{sk}}(c) \cdot a = \mathsf{Dec}_{\mathsf{sk}}(\mathbf{F}_c^m \cdot \mathbf{a} + \mathbf{f}_c^m)$ given the randomness in c is small enough. This a is a plaintext value, so we require that the result be decryptable with a normal secret key. Observe that publicly revealing a value $\mathbf{F}_c^m \cdot \mathbf{a} + \mathbf{f}_c^m$ may leak information on **a**. We therefore *drown* the noise by adding new encryptions $\mathsf{Enc}_{\mathsf{pk}}(0)$ with noise bound β_m in the protocol. The details on the choice of all these parameters depend on the implementation of the protocols and are not discussed any further here. Similarly as above, we will require some additive homomorphism for a small number of additions of ciphertexts obtained from multiplication with a constant. This property follows from the linearity of the procedure for a suitable choice of parameters. An overview over the parameters and notation in this appendix can be found in Fig. B.1.

B.2 Making the Protocols UC-secure

Our proof technique crucially relies upon the simulator being able to extract witnesses from the ZK proofs by rewinding. Unfortunately, such rewinding is not possible in the UC framework. The standard workaround is to base the security on the simulator having other means for obtaining these values (e.g. having secret keys for some encryption scheme or a trapdoor for commitments). One then claims that a distinguisher between those two worlds exist and this distinguisher itself can then do rewinding (but will apparently not have access to the secret information of the simulator). In our case it is obvious that such a proof technique must fail, since we are not aware of trapdoors for our defined commitment scheme.

To make Π_{KG} UC-secure, we use the strengthened definition for the cryptosystem D = (KG, Enc, Dec) and make the additional (mild) setup assumption³ that each party P_i has a key pair $(\overline{pk}_i, \overline{sk}_i)$ with publicly known \overline{pk}_i .

The key generation protocol $\Pi_{\text{KG},\text{UC}}$ follows the same outline as Π_{KG} , with the following difference: the seed s_i is sampled by party P_i in a special procedure **ProEncCommit** where it also generates an encryption $[\![s_i]\!]$ under its key $\overrightarrow{\mathsf{pk}}_i$. P_i will publish the ciphertext and prove that it was computed correctly from s_i and some chosen randomness e using the zero-knowledge proofs from the previous section. The simulator holds the keys $\overrightarrow{\mathsf{sk}}_i$ of the dishonest parties, is able to decrypt each ciphertext and can then send this value to \mathcal{F}_{KGD} as before. The protocol can be found in Fig. B.2. A similar transformation can also be applied to Π_{DEC} and the remaining protocols from this section.

B.3 Computing Powers of the Secret Key

We can moreover use the additional assumptions made on D to allow the computation of powers of the key sk securely. It may first seem counter-intuitive why one would want to compute such a value, but the reason lies in potential homomorphic properties of D:

³ Implicitly, in our protocol we further assume that $\ell_{\mathsf{pk}} = \ell_{\mathsf{sk}} = N$ to be able to encrypt public and private keys. This can easily be generalized, and we just make this assumption to enable a simpler exposition.

Protocol $\Pi_{KG,UC}$

Procedure ProEncCommit(*i*):

- P_i locally samples s ← U^{ℓ_s}_{β_s} as well as e ← U^{ℓ_e}_{β_e} and computes F^{p_k}_e, F^{m_k}_e, F^{m_k}_e.
 P_i computes and broadcasts the commitments

$$C(\boldsymbol{s}), C(\boldsymbol{e}), C(\boldsymbol{F}_{e}^{\overline{\mathsf{pk}}_{i}}\boldsymbol{e}), C(\boldsymbol{F}_{e}^{m}\boldsymbol{s}) \text{ and } C(\boldsymbol{F}_{e}^{\overline{\mathsf{pk}}_{i}}\boldsymbol{e} + \boldsymbol{F}_{e}^{m}\boldsymbol{s})$$

as well as $\llbracket \boldsymbol{s} \rrbracket = \boldsymbol{F}_{e}^{\overline{\mathsf{pk}}_{i}} \boldsymbol{e} + \boldsymbol{F}_{e}^{m} \boldsymbol{s}.$

- 3. Each P_i uses the following proofs towards all parties. Sample the challenge using $\mathcal{F}_{\text{RAND}}$:
 - (a) Π_{BOUND} on C(s) to show that $||s||_{\infty} \leq \beta_s$.
 - (b) Π_{BOUND} on C(e) to show that $||e||_{\infty} \leq \beta_e$.
 - (c) Π_{LIN} on $C(\boldsymbol{e}), C(\boldsymbol{F}_{e}^{\overline{\text{pk}}_{i}} \boldsymbol{e})$ using $\boldsymbol{F}_{e}^{\overline{\text{pk}}_{i}}$. (d) Π_{LIN} on $C(\boldsymbol{s}), C(\boldsymbol{F}_{e}^{m}\boldsymbol{s})$ using \boldsymbol{F}_{e}^{m} .

 - (e) Π_{SUM} on $C(\boldsymbol{F}_{e}^{\overline{\mathsf{pk}}_{i}}\boldsymbol{e}), C(\boldsymbol{F}_{e}^{m}\boldsymbol{s}), C(\boldsymbol{F}_{e}^{\overline{\mathsf{pk}}_{i}}\boldsymbol{e} + \boldsymbol{F}_{e}^{m}\boldsymbol{s}).$
 - (f) $\Pi_{\text{OPEN-X}}$ on $C(\mathbf{F}_{e}^{\overline{\mathbf{p}k}_{i}}\mathbf{e} + \mathbf{F}_{e}^{m}\mathbf{s})$ to show that it opens to $[\![\mathbf{s}]\!]$.
 - If any of the proofs fails, then abort.
- 4. Return $C(\boldsymbol{s})$.

Key Generation:

- 1. Each P_i runs $(s_i, C(s_i)) \leftarrow \text{ProEncCommit}(i)$.
- 2. Each P_i computes and broadcasts the commitments $C(\mathbf{F}_a^{\mathsf{pk}} \mathbf{s}_i), C(\mathbf{F}_a^{\mathsf{sk}} \mathbf{s}_i)$.
- 3. Each P_i uses the following proofs towards all parties. Sample the challenge using \mathcal{F}_{RAND} :
 - (a) Π_{LIN} on $C(\boldsymbol{s}_i), C(\boldsymbol{F}_a^{\text{pk}}\boldsymbol{s}_i)$ using $\boldsymbol{F}_a^{\text{pk}}$.
 - (b) Π_{LIN} on $C(\boldsymbol{s}_i)$, $C(\boldsymbol{F}_a^{\text{sk}}\boldsymbol{s}_i)$ using $\boldsymbol{F}_a^{\text{sk}}$.
 - If one of the proofs fails then abort.
- 4. Denote with \mathbf{pk}_i the committed value in $C(\mathbf{F}_a^{\mathbf{pk}}\mathbf{s}_i)$. Each P_i proves to all parties that $C(\mathbf{F}_{a}^{\mathsf{pk}}\mathbf{s}_{i})$ contains pk_{i} using $\Pi_{\mathsf{OPEN-X}}$. If one of the proofs fails, then abort.
- 5. If all proofs were correct, then output $\mathsf{pk} = \sum_{i \in [n]} \mathsf{pk}_i$.

Fig. B.2. $\Pi_{\text{KG,UC}}$: Protocol for actively secure key generation with UC security.

- The encryption scheme due to [BV11] has an inherent ciphertext growth due to multiplications. The actual key that is used in decryption consists of powers of the secret key. To allow distributed decryption, a sharing of such a power of a secret key must be computed.
- The [BGV12] cryptosystem uses a key switching matrix to cope with the ciphertext growth of [BV11], but this matrix is computed as an encryption of sk^2 (times some constant).

This task of computing a power of the secret key can be achieved using our commitment scheme, its protocols and D. In a proof of security for our protocol, one would have to make an additional assumption on D, namely that it is KDMsecure [BRS02].

Here is how the protocol $\Pi_{\text{KeySQUARED}}$ works on an intuitive level: first, observe that there already are commitments to each \mathbf{sk}_i from Π_{KG} . These commitments can be used in a first step to compute an encryption $c = \text{Enc}_{pk}(\mathbf{sk})$ of the secret key under its public key. This is possible because we can encrypt values that were contained in a commitment into correct ciphertexts, something which we already did in $\Pi_{\text{KG,UC}}$. Therefore, P_i will encrypt \mathbf{sk}_i and prove correctness of the ciphertext. It is safe to reveal this encryption due to the KDM assumption on the cryptosystem. After this is done, these ciphertexts can be added up to obtain c.

Now observe that each share sk_i can be considered as a plaintext element, so we can multiply them with c. This can be done if we compute the matrices $\mathbf{F}_c^m, \mathbf{f}_c^m$ which must exist by assumption on the cryptosystem. These matrices are public and applied to each $C(\mathsf{sk}_i)$ individually, where each P_i knows the correct value that opens the resulting commitment. Before opening it, each P_i will rerandomize the resulting ciphertext such as to hide the share sk_i . The result of the protocol as depicted in Fig. B.3 is then an encryption of sk^2 under pk .

B.4 An Alternative Solution to Distributed Decryption

We want to point out that an alternative approach for distributed decryption can be based on *optimistic decryption*, where the zero-knowledge proofs for the commitments are only executed in the case of a discovered decryption failure (to uncover a dishonest party). During a regular protocol run we will rely on proofs of plaintext knowledge for the ciphertexts which can be amortized using e.g. the technique from [BDLN16]. The reliable decryption technique is similar to [LSSV16], but we moreover allow to identify the cheater.

The optimistic decryption requires that D is somewhat homomorphic. We require that there exists an algorithm \otimes that allows to multiply ciphertexts in a way that allows decryption using D.Dec. Such an algorithm can be realized using the output of $\Pi_{\text{KeySQUARED}}$.

Definition 4 (Multiplicative Property). A distributed cryptosystem D is said to have the multiplicative property if there exists a deterministic $poly(\lambda)$ -time algorithm \otimes such that

$$\Pr\left[\left. m \neq a \cdot b \right| \begin{pmatrix} \mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KG}(1^{\lambda}) \land a, b \in R_p \land \\ c_a \leftarrow \mathsf{Enc}_{\mathsf{pk}}(a) \land c_b \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b) \land \\ c \leftarrow c_a \otimes c_b \land m \leftarrow \mathsf{Dec}_{\mathsf{sk}}(c) \end{pmatrix} \right] \leq \mathsf{negl}(\lambda),$$

where the randomness is taken over⁴ the choice of inputs for KG, Dec, Enc.

Similarly as for $\Pi_{\text{KeySQUARED}}$ we will not prove the security of the protocol, but give some intuition on how it works: to decrypt a ciphertext $[\![x]\!]$ the parties first generate an encryption of a uniformly random value a. They then *encode* xby computing the product $[\![b]\!] = [\![x]\!] \otimes [\![a]\!]$. Before decrypting all three ciphertexts

⁴ To ease of readability, we leave out the full specification of the inputs to KG, Dec but simply assume that they are correct according to \mathcal{F}_{KGD} .

Protocol $\Pi_{KeySquared}$

We assume that a commitment $C(\mathsf{sk}_i)$ of each secret key share is available from Π_{KG} and that $||\mathsf{sk}_i||_{\infty} < p$.

1. Each P_i samples $\boldsymbol{v}_i \stackrel{\$}{\leftarrow} \mathcal{U}_{\beta_e}^{\ell_e}$ and computes and broadcasts the commitments

 $C(\boldsymbol{v}_i), C(\boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{v}_i), C(\boldsymbol{F}_e^m\mathsf{sk}_i), C(\boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{v}_i + \boldsymbol{F}_e^m\mathsf{sk}_i) \text{ as well as } [\![\mathsf{sk}_i]\!] = \boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{v}_i + \boldsymbol{F}_e^m\mathsf{sk}_i$

- 2. Each P_i uses the following proofs towards all parties. Sample the challenge using $\mathcal{F}_{\text{RAND}}$:
 - (a) Π_{BOUND} on $C(\boldsymbol{v}_i)$ to show that $||\boldsymbol{v}_i||_{\infty} \leq \beta_e$.
 - (b) Π_{LIN} on $C(\boldsymbol{e}_i), C(\boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{v}_i)$ using $\boldsymbol{F}_e^{\mathsf{pk}}$.
 - (c) Π_{LIN} on $C(\mathsf{sk}_i)$, $C(F_e^m\mathsf{sk}_i)$ using F_e^m .
 - (d) Π_{Sum} on $C(\boldsymbol{F}_{e}^{\mathsf{pk}}\boldsymbol{v}_{i}), C(\boldsymbol{F}_{e}^{m}\mathsf{sk}_{i}), C(\boldsymbol{F}_{e}^{\mathsf{pk}}\boldsymbol{v}_{i} + \boldsymbol{F}_{e}^{m}\mathsf{sk}_{i}).$
 - (e) $\Pi_{\text{OPEN-X}}$ on $C(\boldsymbol{F}_{e}^{\mathsf{pk}}\boldsymbol{v}_{i} + \boldsymbol{F}_{e}^{m}\mathsf{sk}_{i})$ to show that it opens to $[\![\mathsf{sk}_{i}]\!]$. If one of the proofs fails then abort.
- 3. Each P_i locally computes $[sk] = \sum_{j=1}^{n} [sk_j]$ and F_c^m, f_c^m from [sk].
- 4. Each P_i samples $\boldsymbol{w}_i \stackrel{\$}{\leftarrow} \mathcal{U}_{\beta_m}^{\ell_e}$ and computes and broadcasts the commitments

 $C(\boldsymbol{w}_i), C(\boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{w}_i), C(\boldsymbol{F}_c^{m}\mathsf{sk}_i + \delta_{1i} \cdot \boldsymbol{f}_c^{m}), C(\boldsymbol{F}_c^{m}\mathsf{sk}_i + \delta_{1i} \cdot \boldsymbol{f}_c^{m} + \boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{w}_i),$

as well as

$$\llbracket \mathsf{sk} \cdot \mathsf{sk}_i \rrbracket = \boldsymbol{F}_c^m \mathsf{sk}_i + \delta_{1i} \cdot \boldsymbol{f}_c^m + \boldsymbol{F}_e^{\mathsf{pk}} \boldsymbol{w}_i$$

where δ_{xy} is the Kronecker Delta function.

- 5. Each P_i uses the following proofs towards all parties. Sample the challenge using $\mathcal{F}_{\text{RAND}}$:
 - (a) Π_{BOUND} on $C(\boldsymbol{w}_i)$ to show that $||\boldsymbol{w}_i||_{\infty} \leq \beta_m$.
 - (b) Π_{LIN} on $C(\boldsymbol{w}_i), C(\boldsymbol{F}_e^{\mathsf{pk}} \boldsymbol{w}_i)$ using $\boldsymbol{F}_e^{\mathsf{pk}}$.
 - (c) Π_{LIN} on $C(\mathsf{sk}_i), C(\mathbf{F}_c^m \mathsf{sk}_i + \delta_{1i} \cdot \mathbf{f}_c^m)$ using the linear function $g(x) = \mathbf{F}_c^m x + \delta_{1i} \cdot \mathbf{f}_c^m$.
 - (d) Π_{SUM} on $C(\boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{w}_i), C(\boldsymbol{F}_c^m\mathsf{sk}_i + \delta_{1i} \cdot \boldsymbol{f}_c^m), C(\boldsymbol{F}_c^m\mathsf{sk}_i + \boldsymbol{f}_c^m + \boldsymbol{F}_e^{\mathsf{pk}}\boldsymbol{w}_i).$
 - (e) $\Pi_{\text{OPEN-X}}$ on $C(\boldsymbol{F}_c^m \mathsf{sk}_i + \delta_{1i} \cdot \boldsymbol{f}_c^m + \boldsymbol{F}_e^{\mathsf{pk}} \boldsymbol{w}_i)$ to show that it opens to $[\![\mathsf{sk} \cdot \mathsf{sk}_i]\!]$. If one of the proofs fails then abort.
- 6. Each P_i locally computes $[sk^2] = \sum_{j=1}^n [sk \cdot sk_j]$. Output $[sk^2]$.

Fig. B.3. $\Pi_{\text{KeySQUARED}}$: Protocol for actively secure generation of powers of secret keys.

unreliably, each P_i commits to the values $\mathbf{F}_x \mathbf{sk}_i, \mathbf{e}_i^x, \mathbf{d}_i^x$ that it will use in Dec to decrypt $[\![x]\!]$, as well as those values used in the decryption of $[\![a]\!], [\![b]\!]$. Thereafter, the parties unreliably decrypt $[\![x]\!], [\![a]\!], [\![b]\!]$ by opening the commitments to $\mathbf{d}_i^x, \mathbf{d}_i^a$ and \mathbf{d}_i^b . All parties check that $a \cdot x = b$.

If this equality holds, then we consider the result as correct. If, on the other hand, it does not hold, then each party proves in zero knowledge that its d_i^x, d_i^a, d_i^b were correctly generated (as in a correct decryption procedure) based on the commitments that it provided. The protocol is presented in Fig. B.4, where Step (1) and Step (2) can be done ahead of decryption time.

Protocol Π_{DecAlt}

A protocol to decrypt a ciphertext [x].

- Each P_i samples a_i ^{\$} ⊂ R_p uniformly at random and computes [[a_i]] ← Enc_{pk}(a_i).
 Each P_i broadcasts [[a_i]] together with a proof of plaintext knowledge for Enc.
- 3. Each P_i locally computes $\llbracket a \rrbracket = \sum_{i \in [n]} \llbracket a_i \rrbracket$ and $\llbracket b \rrbracket \leftarrow \llbracket x \rrbracket \otimes \llbracket a \rrbracket$.
- Each P_i locally samples randomness e^x_i, e^a_i, e^b_i [♣] U^{ℓg}_{βa} and computes F_x as used in Dec to decrypt [[x]] as well as F_a for [[a]] and F_b for [[b]]. Then compute

$$d_i^x = F_x \mathsf{sk}_i + e_i^x$$
 and $d_i^a = F_a \mathsf{sk}_i + e_i^a$ and $d_i^b = F_b \mathsf{sk}_i + e_i^b$.

5. Each P_i broadcasts

$$C(\boldsymbol{F}_{x}\mathsf{sk}_{i}), C(\boldsymbol{F}_{a}\mathsf{sk}_{i}), C(\boldsymbol{F}_{b}\mathsf{sk}_{i}), C(\boldsymbol{e}_{i}^{x}), C(\boldsymbol{e}_{i}^{a}), C(\boldsymbol{e}_{i}^{b}) \text{ and } C(\boldsymbol{d}_{i}^{x}), C(\boldsymbol{d}_{i}^{b}).$$

- 6. Each P_i generates auxiliary commitments to commit to d_i^x, d_i^a, d_i^b towards all parties.
- 7. Each P_i opens the auxiliary commitments to d_i^x, d_i^a, d_i^b .
- 8. All parties check that

$$\mathsf{decode}(\sum_{i\in[n]}\boldsymbol{d}_i^x)\cdot\mathsf{decode}(\sum_{i\in[n]}\boldsymbol{d}_i^a)=\mathsf{decode}(\sum_{i\in[n]}\boldsymbol{d}_i^b)$$

If yes, then they output $x \leftarrow \mathsf{decode}(\sum_{i \in [n]} \mathbf{d}_i^x)$ and terminate.

- 9. Otherwise, for each $i \in [n]$ the parties do the following, where all parties abort with P_i if a check fails:
 - (a) P_i proves using Π_{BOUND} that $C(\boldsymbol{e}_i^x), C(\boldsymbol{e}_i^a), C(\boldsymbol{e}_i^b)$ have ∞ -norm at most β_d .
 - (b) P_i proves using Π_{LIN} that $C(\boldsymbol{F}_x \mathsf{sk}_i), C(\boldsymbol{F}_a \mathsf{sk}_i), C(\boldsymbol{F}_b \mathsf{sk}_i)$ are derived from $C(\mathsf{sk}_i)$ using $\boldsymbol{F}_x, \boldsymbol{F}_a, \boldsymbol{F}_b$.
 - (c) P_i runs Π_{SUM} on the tuples $- (C(\boldsymbol{F}_x \mathsf{sk}_i), C(\boldsymbol{e}_i^x), C(\boldsymbol{d}_i^x))$ $- (C(\boldsymbol{F}_a \mathsf{sk}_i), C(\boldsymbol{e}_i^a), C(\boldsymbol{d}_i^a))$ $- (C(\boldsymbol{F}_b\mathsf{sk}_i), C(\boldsymbol{e}_i^b), C(\boldsymbol{d}_i^b)).$ (d) P_i proves using $\Pi_{\text{OPEN-X}}$ that $C(\boldsymbol{d}_i^x), C(\boldsymbol{d}_i^a), C(\boldsymbol{d}_i^b)$ open to $\boldsymbol{d}_i^x, \boldsymbol{d}_i^a, \boldsymbol{d}_i^b$.

Fig. B.4. Π_{DecAlt} : Alternative protocol for the decryption of ciphertexts.