Asymptotically-Good Arithmetic Secret Sharing over $\mathbb{Z}/p^\ell\mathbb{Z}$ with Strong Multiplication and Its Applications to Efficient MPC*

Ronald Cramer[1,2], Matthieu Rambaud[3], and Chaoping Xing[4,5]

[1] Cryptology Group, CWI Amsterdam
[2] Leiden University
[3] Telecom Paris, Institut polytechnique de Paris
[4] School of Electronic Information and Electric Engineering, Shanghai Jiao Tong University
[5] State Key Laboratory of Cryptology, Beijing, China

Abstract. This paper studies information-theoretically secure multiparty computation (MPC) over rings $\mathbb{Z}/p^\ell\mathbb{Z}$. In the work of [Abs+19a, TCC'19], a protocol based on the Shamir secret sharing over $\mathbb{Z}/p^\ell\mathbb{Z}$ was presented. As in the field case, its limitation is that the share size grows as the number of players increases. Then several MPC protocols were developed in [Abs+20, Asiacrypt'20] to overcome this limitation. However, (i) their offline multiplication gate has super-linear communication complexity in the number of players; (ii) the share size is doubled for the most important case, namely over $\mathbb{Z}/2^\ell\mathbb{Z}$ due to infeasible lifting of self-orthogonal codes from fields to rings; (iii) most importantly, the BGW model could not be applied via the secret sharing given in [Abs+20, Asiacrypt'20] due to lack of strong multiplication.

In this paper we overcome all the drawbacks mentioned above. Of independent interest, we establish an arithmetic secret sharing with strong multiplication, which is the most important primitive in the BGW model. Incidentally, our solution to (i) has some advantages over the concurrent one of [PS21, EC'21], since it is direct, is only one-page long, and furthermore carries over $\mathbb{Z}/p^\ell\mathbb{Z}$. Finally, we lift Reverse Multiplication Friendly Embeddings (RMFE) from fields to rings, with same (linear) complexity. Note that RMFE has become a standard technique for communication complexity in MPC in the regime over many instances of the same circuit, as in [Cas+18, Crypto'18] and [DLN19, Crypto'19]. We thus recover the same amortized complexity of MPC over $\mathbb{Z}/2^\ell\mathbb{Z}$ than over fields.

To obtain our theoretical results, we use the existence of lifts of curves over rings, then use the known results stating that Riemann-Roch spaces are free modules. To make our scheme practical, we start from good algebraic geometry codes over finite fields obtained from existing computational techniques. Then we present, and implement, an efficient algorithm to Hensel-lift the generating matrix of the code, such that the multiplicative conditions are preserved over rings. On the other hand, a random lifting of codes over rings does not preserve multiplicativity in general. Finally we provide efficient methods for sharing and reconstruction over rings.

* Supported by Horizon 2020 74079(ALGSTRONGCRYPTO) and NSFC under grant 12031011 and the National Key Research and Development Project 2020YFA0712300

1 Introduction

MPC over rings $\mathbb{Z}/p^\ell\mathbb{Z}$, is a model relevant for secure computation of functions which are naturally expressed over rings of integers $\mathbb{Z}/p^\ell\mathbb{Z}$. The most important case is $\mathbb{Z}/2^\ell\mathbb{Z}$ with ℓ (a multiple of) the length of machines integers. In this model, the computation complexity is counted in terms of elementary additions and multiplications in $\mathbb{Z}/p^\ell\mathbb{Z}$, and the communication complexity is the number of elements of $\mathbb{Z}/p^\ell\mathbb{Z}$ sent. By contrast, the previous model of MPC are arithmetic circuits in $\mathbb{F}_p$. But computations modulo p are not natively done by processors. Unless $p = 2$, which is the case studied by MPC for the functions expressed naturally as binary circuits. It appears from the literature that emulating MPC over the integers, from MPC in $\mathbb{F}_p$, incurs a substantial overhead in complexity. For instance, the protocol of [Dam+06] for bit decomposition of numbers modulo a large p, in order to perform secure comparisons, costs $\log(p)\log(\log(p))$ secure multiplications modulo p. Whereas comparisons directly between integers modulo a power of 2 are much more efficient ([Ara+18]).

1.1 Related works

In a recent line of work on efficient MPC over $\mathbb{Z}/p^\ell\mathbb{Z}$, significant advances have been made in order to avoid the overhead incurred by this emulation, by redesigning basic arithmetic MPC so as to work "more directly" over the ring in question. The first published paper [Cra+18] in this line introduces the SPD$\mathbb{Z}_{2^k}$ protocol, a full redesign of the well-known SPDZ-protocol [Dam+12], the benchmark for the case of cryptographic security with dishonest majority in Beaver's preprocessing model, that works directly over the rings in question and that is essentially as efficient as the most efficient SPDZ-incarnation. See also the compilers of [DOS18; Abs+19b] from passive security over rings to active security over rings. For more discussion about practical advantages, see [Cra+18] and its follow-up [Dam+19], which also reports on applications to machine-learning that significantly outperform approaches from field-based MPC. Maliciously secure machine learning directly over the integers is now becoming the standard (e.g. [PS20]).

Closer to us is the line of work [Abs+19a; Abs+20], that aims at answering the question if information theoretically secure MPC over $\mathbb{Z}/p^\ell\mathbb{Z}$, has complexity equal to the one of MPC over $\mathbb{F}_p$. The issue is simple: suppose that one has the choice between two protocols with the same complexities: measured over $\mathbb{F}_p$ for the former, and over $\mathbb{Z}/p^\ell\mathbb{Z}$ for the latter. Then the latter protocol is automatically the most efficient to securely compute any function over $\mathbb{Z}/p^\ell\mathbb{Z}$, since no emulation is needed. The present paper firstly addresses this question mainly in the plain model [CDN15, §5], denoted "BGW", that is: assuming only authenticated channels and requiring perfect security. So in particular, we have that the number of malicious corruptions is $t < n/3$, since no broadcast channel is available beyond this bound in the BGW model. [Abs+19a] considers MPC in the BGW model over rings. It adapts the protocol of [BH08], including the

secret sharing over $\mathbb{Z}/p^\ell\mathbb{Z}$ adapted from Shamir, and thus inherits the amortized suboptimal $O(n\log(n))$ communication complexity of [BH08].

We then consider the setting of Rabin&Ben-Or, denoted as "honest majority", which assumes a broadcast and requires unconditional statistical security, tolerating $t < n/2$ corruptions. Until recently the best amortized communication complexity over fields was [BFO12], in $n\log(n)$, plus a term in n^2 times the depth of the circuit. Let us first discuss the n^2 term, which was removed over fields by [GSZ20]. As noticed in [Abs+20] (at the beginning of §6), the main tool of [GSZ20] is the Batched Triple Sacrifice protocol of [BFO12], that checks correctness of shared Beaver triples. It runs in the offline phase, and has $O(n\log(n))$ communication complexity (to be sure, the notation ϕ in [GSZ20] stands for the log of the size of the field, it is required to be $\phi \geq \log(n)$). This Batched Triple Sacrifice was then carried over rings in [Abs+20], resulting in an overall amortized communication complexity in $O(n\log(n))$ also over rings.

Let us now discuss the $\log(n)$ overhead. Concurrently to our paper, it has been removed by [PS21, EC'21], over fields. Their technique use as black box the RMFE with optimal rate of [Cas+18], which come from algebraic curves.

In addition to the above super-linear offline communication complexity, there are some other drawbacks in [Abs+20]: (i) their secret sharing schemes with (standalone) Multiplication are constructed with a double sharing, which thus doubles the size (ii) their way around this doubling, only for $p \geq 3$, uses asymptotically good families of self-dual codes, for which no practical construction is known (by contrast with good families of codes from algebraic curves / function fields, whose computation is widely studied); (iii) most importantly, the secret sharing scheme given in [Abs+20] cannot be adopted for the BGW model due to lack of Strong multiplication.

1.2 Our focus

Our main focus are the two fundamental primitives for MPC in the BGW model. We also deal with the asymptotic complexity of MPC under honest majority, especially the Batched Triple Sacrifice which costs the so-far $\log(n)$ communication overhead. We are finally concerned by the computational efficiency of general reconstruction methods of linear secret sharing schemes over rings, which was not dealt with at all in previous works.

The first primitive for MPC in the BGW model is *arithmetic secret sharing with Strong multiplication* (ASSSM). Recall that such a scheme with respect to adversary bound t, guarantees both: secrecy from any t shares, and, reconstruction of the product of two shared secrets, from any list of $n - t$ products of pairwise shares of these secrets. By contrast, secret sharing with (standalone) Multiplication only, requires *all* the n pairwise products of shares for reconstruction of the product. The simplest example is the Verifiable secret sharing of BGW itself. Importance of Strong multiplication is formalized in the Theorem 3 of [CDM00], as the building block of error-free MPC protocols. Namely, it is emphasized in [CDN15, p114] as the tool enabling not to restart the execution of the protocol, even when a player openly misbehaves. Notice that the $\log(n)$

overhead is inherited from Shamir's secret sharing, which operates in finite fields of cardinality *at least as large* as the number of players. This limitation was removed in the series of papers [CC06; Cas+09; CCX11] using algebraic geometric codes over fields. Notice that these state of the art ASSSM and constant size of shares, motivated the "MPC in the head technique" [Ish+07], see [Cas16, §5] for other applications. In this paper we ask if the same tight size of shares is achievable over rings. We also ask if the same efficiency of constructions is achievable as over fields [Hes02; Khu04; Shu+01; SG20]. Also, much optimisation has been made for sharing/reconstruction algorithms over fields [SW99; GS99; NW17].

The second primitive are *Reverse multiplication friendly embeddings* (RMFE). They enable to emulate several circuits in parallel over small finite fields $\mathbb{F}_p$, from a single circuit over a large extension $\mathbb{F}_{p^m}$. They are introduced in [Cas+18], and are the main tool for the upper bounds of [DLN19; BMN18; DLS20; CG20; PS21]. RMFE enable to linearize the amortized communication complexity of perfectly secure MPC, over multiple instances of the same circuit (with possibly different inputs), while preserving an *optimal* corruption tolerance. Recall that a RMFE [Cas+18, Definition 1] (recalled in §5.4), is an embedding from some vector space $\mathbb{F}_q^k$ over $\mathbb{F}_q$, into some field extension $\mathbb{F}_{q^m}$, which "carries" the multiplication in $\mathbb{F}_{q^m}$ into the component-wise multiplication of vectors in $\mathbb{F}_q^k$ (the same one as for multiplicative secret sharing). The larger the ratio k/m, the better the complexity of MPC is amortized. Again, RMFE with polynomial encoding (as in Shamir secret sharing) exist up to $k \leq q+1$. And again, this limit of the field size was removed in [Cas+18, Theorem 3] with constructions from algebraic geometry coding. Namely, they achieve for any fix q, a slowly growing infinite family of parameters k, m such that the ratios k/m are lower bounded by a *constant*, which is optimal. We thus ask if the same ratios are achievable over rings, and if constructions are as efficient.

1.3 Our contributions

1.3.1 Asymptotically optimal Strong multiplication over rings

Main Theorem 1. For every p and ℓ, for any fixed even r larger than some $\hat{r}(p)$, we have a slowly growing infinite family of number of integers n, such that there exists an ASSSM over the fixed ring $\mathbb{Z}/p^\ell\mathbb{Z}$, with n shares, with constant size of shares r and t-adversary bound such that $1/3 - t/n > 0$ a constant arbitrarily close to 0 (in $O(p^{-r/2})$).

More precisely and generally: all parameters $(n, p, \hat{r}(p), t)$ published in [CC06; Cas+09] over fields $\mathbb{F}_p$, *also* hold over rings. We have stronger than privacy: *uniformity* of the projection on any t shares of the space of vectors of shares of any given fixed secret. Moreover, the scheme obtained by reduction modulo p may be assumed to be asymptotically good as well. [6] Last but not least, sharing and reconstruction over $\mathbb{Z}/p^\ell\mathbb{Z}$ have the *same computational complexity* than for ASSSM over fields $\mathbb{F}_p$.

[6] This fact is quite useful in some practical protocol applications but it is not strictly necessary for general arithmetic MPC

This thus closes the gap between the complexity of ASSSM over fields, and over rings. Since this result is tight, we do not further justify why our construction uses "Galois rings" extensions as an intermediary step. Although we hope that it will be clear from §1.4, §2.1 (and also [Abs+19a; Abs+20]) that these objects play the same auxiliary role over rings, as finite fields extensions do over fields. Concretely, under the hood is that $\mathbb{F}_p$ is embedded into $\mathbb{F}_{p^r}$ in order to access ASSSM/RMFE with good properties, which are then lifted over Galois ring extensions, then seen as free modules over $\mathbb{Z}/p^\ell\mathbb{Z}$. But for simplicity, we refrained from stating that the above theorem also holds for any Galois ring extension of degree r, with the same parameters $(n, p, r(p), t)$ than [CC06; Cas+09] over $\mathbb{F}_{p^r}$. We also kept simple the formula and made explicit only the case where t is close to $n/3$. To be sure, the parameters of [Cas+09] also enable smaller sizes of shares, at the cost of a lower t (using multiplication friendly embeddings). The last claim, on efficiency, will follow from the algorithms of Theorem 5 below. Technically, Theorem 5 applies here since the componentwise squares (see §3.1) of the codes constructed are included in "free codes" of dimension as small as the codes of the ASSSM over finite fields.

1.3.2 Optimal communication of MPC under honest majority. We remove the so-far aforementioned amortized $\log(n)$ communication overhead, which also held over fields. The bottleneck comes from the offline phase, in the subprotocol of [BFO12] checking triples, e.g., as transposed over rings in [Abs+20, §6.6]. Recall that the baseline method of [BFO12] proceeds by encoding many triples in three polynomials, then succinctly check the multiplicative relation between these polynomials. We start by replacing it by an alternative construction of Batched Triple Sacrifice over fields of fixed size. This construction is closely related to the strong multiplication property, it is stated and proven in Proposition 16. It enables to recover the main result of [PS21] in only one page. We then lift the construction over rings with the same methods as before.

Main Theorem 2. In the model of [RB89]: honest majority and assuming broadcast, then there exists a statistically secure MPC protocol with guaranteed output delivery and amortized communication complexity (both online and offline) linear in the number of players per multiplication gate.

Anticipating on the next result, notice that we could also have directly lifted over rings the construction of [PS21]. Indeed, theirs is based on the RMFE of [Cas+18], which we lift over rings with the same asymptotic rates.

1.3.3 Amortized complexity of MPC over rings. We construct, in §5.2 an infinite family of RMFE over rings with same *constant* asymptotic ratio as the ones of [Cas+18]. Combined with the tight complexities of general LSSS proven in Theorem 5, this enables to carry over rings the results of [Cas+18] with the same computational and communication complexities:

Main Theorem 3. In the BGW-model, there is an efficient MPC protocol for n parties secure against the maximal number of active corruptions $t < n/3$ that computes $\Omega(\log n)$ evaluations of a single circuit over $\mathbb{Z}/p^\ell\mathbb{Z}$ in parallel with an amortized communication complexity (per instance) of $O(n)$ elements of $\mathbb{Z}/p^\ell\mathbb{Z}$ per gate, and same computational complexity than in [Cas+18, Thm 1 & 2]. Combining with the Franklin-Yung paradigm [FY92], we get:

In the BGW-model, for every $\epsilon > 0$, there is an efficient MPC protocol for n parties secure against a submaximal number of active corruptions $t < (1-\epsilon)n/3$ that computes $\Omega(n \log n)$ evaluations of a single circuit over $\mathbb{Z}/p^\ell\mathbb{Z}$ in parallel with an amortized communication complexity (per instance) of $O(1)$ elements of $\mathbb{Z}/p^\ell\mathbb{Z}$ per gate.

1.3.4 Optimal share sizes and computability under honest majority

The asymptotically good ASSSM of Theorem 1 have a fortiori standalone Multiplication. So they can be used as a replacement for the schemes constructed in [Abs+20, §4.1]. Especially for $p = 2$, recall that Multiplication of their schemes is obtained via a double sharing, which thus doubles the size of the shares (as stressed in the roadmap of [Abs+20, §3]). Our construction thus *divides their sizes of shares by* 2 *for* $p = 2$. A corollary of above, is that the active protocol presented in Section 6 of [Abs+20], which requires standalone Multiplication, now works with share sizes reduced to half, and now using *computable* families of codes, including from AG/function fields.

1.3.5 Practical computability (continued)

Main Theorems 1, 2 and 3 rely on objects (ASSSM or RMFE or the related ones of Proposition 16) with good asymptotic properties of which we prove existence. We then describe in §4 efficient algorithms to construct these objects.

Theorem 4. *Starting from any ASSSM over any fixed field* $\mathbb{F}_p$ *considered in [CC06; Cas+09; CCX11], then, obtaining the lifts over* $\mathbb{Z}/p^\ell\mathbb{Z}$ *for any* ℓ*, as predicted by Main Theorem 1, boils down to solving* ℓ *instances of a* linear system *over* $\mathbb{F}_p$ *with* $\Omega(n^6)$ *coefficients. Alternatively,* $\log(\ell)$ *linear systems: modulo* $p, p^2, p^4, \ldots, (p^\ell)^{\frac{1}{2}}$.

We have an analogous system to obtain the RMFE predicted by Main Theorem 3, from the ones of [Cas+18, Theorem 5].

A formal description is given in §4 (for ASSSM) and §5.4 (for RMFE), a toy example in §3.1. A proof of Theorem 4 is given in the long version. It requires to prove that AG codes have a free lift whose square is also free, which requires additional methods than those given in §3.3. We illustrate efficiency of our method in §4.1 by lifting a strongly multiplicative secret sharing scheme over $\mathbb{F}_{16}$ for 64 players and adversary threshold $t = 13$, into a scheme over the Galois extension of degree four of $\mathbb{Z}/2^{100}\mathbb{Z}$, in a minute on a single processor.

1.3.6 Tight computational complexity of linear secret sharing schemes (LSSS) over rings

Although theoretical results for error correction over rings are shown in [Abs+19a, Construction 1 & Proposition 1], it is not yet clear in the literature if there exists *effective algorithms* for even the simple task of reconstruction of a secret with only erasures. We fill this gap by providing algorithms for sharing and reconstruction of linear secret sharing schemes (LSSS) over rings that arize from *free* codes. In particular it proves our efficiency claims in the Main Theorems above. A free code C over $\mathbb{Z}/p^\ell\mathbb{Z}$ is by definition the linear span of independent vectors with coordinates in $\mathbb{Z}/p^\ell\mathbb{Z}$, in particular it is of same dimension than its reduction $\overline{C}$ modulo p, which is a code over $\mathbb{F}_p$. In particular, all the ASSSM constructed in this paper have this property, as well as the objects studied in [Abs+20] and in [Abs+19a] (which considers the specific case of Shamir secret sharing over rings). On the other hand, LSSS arising from nonfree codes have bad computational complexity, as we illustrate in Counterexample 10. We provide computational complexities that match the ones over finite fields, so which are *tight*. For simplicity the following theorem is stated over $\mathbb{Z}/p^\ell\mathbb{Z}$, but it will be clear from the proof that it obviously also holds over any Galois ring extension.

Theorem 5. *Let n, ℓ be integers, consider* a free code C *in $(\mathbb{Z}/p^\ell\mathbb{Z})^{n+1}$ and let ψ the corresponding (LSSS) with n shares in $\mathbb{Z}/p^\ell\mathbb{Z}$, such that (without loss of generality) the secret is encoded in the 0-th coordinate of codewords. Denote $\overline{C}$ the code reduced modulo p and $\overline{\psi}$ the corresponding LSSS (which is ψ modulo p) over $\mathbb{F}_p$. We have:*

(A) The task of computing a generating matrix of C in systematic form, from any generating matrix of C and, more generally, Gauss pivot, has same computational complexity as modulo p, plus $O((\dim C)(\log \ell))$.

(A') Then, sharing a secret using ψ (thus of bitsize ℓ times larger) has same computational complexity than using $\overline{\psi}$. As for sharewise multiplication.

(B) Let $I \subset \{1, \ldots, n\}$ be a set of $n - d(\overline{C}) + 2$ indices of shares. Then, there exists a linear *map $\phi_I : (\mathbb{Z}/p^\ell\mathbb{Z})^{|I|} \longrightarrow \mathbb{Z}/p^\ell\mathbb{Z}$ that reconstructs the secret, with the* same complexity *than a reconstruction map $\overline{\phi_I}$ for $\overline{\psi}$. Moreover, ϕ_I can be compiled from a reconstruction map $\overline{\phi_I}$ for the LSSS modulo p, essentially for the cost of one matrix inversion in $(\mathbb{Z}/p^\ell\mathbb{Z})^{|I|\times|I|}$.*

Notice that the matrix inversion required in (B) can be computed using the Gauss pivot of (A).

1.4 Difficulties and intuitions of the constructions

Only algebraic-geometric (AG) constructions such as in [CC06] are so far known to enable ASSSM over fields of constant sizes for an arbitrarily large number of shares. They follow the same pattern than the scheme of Shamir, which is a particular case. First, select an algebraic curve (e.g. all the points in $\mathbb{F}_q$ plus the "point at infinity", in the case of Shamir). Second, select a "Riemann-Roch" vector space of functions (e.g. the polynomials of degree $\leq d$ in the case of Shamir: said

otherwise, the space $\mathcal{L}(d\infty)$ of polynomials "vanishing at order at least d at infinity"). Then, select a particular point P_0 on the curve (e.g. the point 0 in Shamir). To share a secret s, select a function at random in the Riemann-Roch space that evaluates to s at P_0. Then evaluate it on n predefined points of the curve to obtain the shares. In what follows we will instead take a coding-based approach. This has both the advantage to make proofs which are more black box in the AG codes used, and also, our efficient methods will actually directly lift the generating matrices of such AG codes over fields.

For C a code (over a field or a ring), we denote as *componentwise square* C^{*2} the code of same length which is generated by all the products of any two codewords of C component by component. Strong multiplication of the LSSS from C thus requires that C^{*2} has large distance, thus be of small size. The central problem of this paper is thus brought down to: starting from a free code C over a ring (typically a finite field) which has free square of small dimension, then find a code $\widetilde{C}$ in a larger ring, that reduces to $C \bmod p^\ell$, and has square contained in a free code of small size. On the one hand, it is trivial to lift Reed-Solomon (RS) codes over rings, in a way that preserves their remarkably small componentwise square. Indeed, lifts of RS codes are given for free: these are the RS codes over rings. RS codes over rings were studied in [Abs+19a], but, as over finite fields, these RS codes have a $\log(n)$ size overhead. This inefficiency is one of the main motivations of the present paper. On the other hand, when trying to lift AG codes with larger genus, in order to remove this overhead, we hit the main difficulty of this paper. Namely, we illustrate in §3.1 that lifting at random (as done in [Abs+20]) almost certainly *fails* to preserve the small dimension of the square.

But our theoretical results imply that a solution exists, which we are able to compute efficiently. Anticipating on them, we first present a toy example in §3.2. Recall that RS codes are the simplest case of AG codes, namely, over the curve $\mathbf{P}^1$ (the "projective line"), which is of genus 0. This is why our toy example §3.2 illustrates the simplest nontrivial example, which is a curve of genus 1.

To obtain our theoretical results over rings, we first use known theorems that state the existence of lifts of curves over rings. We then apply results of Judy Walker that state that Riemann Roch spaces are free modules, and also, the codes deduced from their injective evaluation at points of the curve. On the other hand, to compute the codes concretely, we will follow a direct approach. Namely, instead of lifting curves over rings, we will directly lift a generating matrix of the code, such that the multiplicative conditions are preserved. On the face of it, there are more constraints than variables. But a result is always returned. Proving this fact requires strictly more than that C lifts with a small square. Indeed, it also requires *freeness* of the square of the lift $\widetilde{C}^{*2}$, which is harder. This is why we prove it in the long version only.

The reader may wonder why we did not directly compute lifts of the curves. The reason is that the theoretical results require that the curves be represented with "smooth" equations, in particular, with many variables. But in practice, good curves are expressed in terms of equations with two variables only. And

there is no efficient method today to compute smooth lifts of such "plane" models, that have many "non smooth" points. Let alone computing Riemann-Roch spaces of smooth models curves over rings, which is out of the scope of existing research (except Walker-Voloch, for smooth curves in the plane).

1.5 Roadmap

In §2 we show that LSSS derived from free codes over Galois rings have same privacy and reconstruction threshold as over the field modulo p. In §2.3 and §2.4 we present efficient sharing and reconstruction algorithms (proof of Theorem 5 (A) (A') and (B)). We show conversely in Counterexample 10 that *there does not exist a linear reconstruction map* for a large class of linear codes over rings which are not free. This is why we focus on LSSS derived from free codes. Let us mention for the hurry reader that the results in §2.3 and §2.5 are not used for the proof of Main Theorem 1.

In §3 we highlight the nontriviality of Main Theorem 1 on a toy example in §3.1, then illustrate in §3.2 how to compute a multiplicative lift of it. We then prove the Theorem in §3.3.

In §4 we elaborate more on the Hensel lifting method illustrated in the toy example.

In §5 we prove the aforementioned applications of the theory to MPC. First with a proof of Proposition 16 (the triples sacrifice algorithm over a field/ring of constant size), then with a proof of Main Theorem 3. The proof involves RMFE over rings with same asymptotically constant rate than over fields: we also describe the effective algorithm to construct them in §5.4.

2 LSSS from *Free* Codes have Optimal Complexity

In §2.1 we introduce Galois ring extensions, and highlight that they the same size and computational overhead over $\mathbb{Z}/p^\ell\mathbb{Z}$, than finite field extensions have over $\mathbb{F}_{p^r}$. The presentation should be self-contained, but the reader can also refer to [Abs+20; Abs+19a]. In §2.2 and §2.3 we consider general LSSS from free codes over rings, and prove the tight complexity claims (A') and (B) of Theorem 5 for sharing and reconstruction. In §2.5 we show that free codes are generated from any lift of any basis. All the basics are recalled, but the reader can alternatively refer to [Abs+20, §2-§4]

2.1 Optimal Complexities in Galois Rings Extensions $R_\ell(r)$

2.1.1 Equal Computational and Sizes for Elementary Operations Let p be a positive prime number and $\mathbb{F}_p := \mathbb{Z}/p\mathbb{Z}$ the finite field. Then, when operating on objects with coordinates in $\mathbb{F}_p$, we say that the *computational complexity* is the number of *elementary operations in* $\mathbb{F}_p$ (where one can possibly weight differently additions, scalar multiplications and bilinear multiplications). Now, $\ell \geq 1$ denoting an integer, the second context encountered in this paper

are objects with coordinates in $\mathbb{Z}/p^\ell\mathbb{Z}$ the ring of integers modulo p^ℓ. In these cases, we say that the computation complexity is the number of *elementary operations in* $\mathbb{Z}/p^\ell\mathbb{Z}$ (where one gives the same weights as before to additions, scalar multiplications and bilinear multiplications). Likewise, the *communication (or size) complexity* is, in the first context: the number of elements in $\mathbb{F}_p$ which are sent by honest players; whereas in the second context it is the number of elements in $\mathbb{Z}/p^\ell\mathbb{Z}$ which are sent.

Galois-rings are defined as follows. Let $r \geq 1$ be a positive integer and $\overline{f(X)} \in \mathbb{F}_p[X]$ a monic irreducible polynomial of degree r. This defines the finite field extension or degree r:

$$\mathbb{F}_{p^r} = \mathbb{F}_p < \delta >:= \mathbb{F}_p[X]/\overline{f(X)}$$

which is a vector space of dimention r over $\mathbb{F}_p$ with basis $1, \delta, \ldots, \delta^{r-1}$ and multiplication rule defined by the multiplication modulo $\overline{f(X)}$. Now, consider any monic polynomial $f(X) \in \mathbb{Z}/p^\ell\mathbb{Z}$ which reduces to $\overline{f(X)}$ modulo p. Then this defines the Galois ring extension of degree r:

$$R_\ell(r) = \mathbb{Z}/p^\ell\mathbb{Z} < \Delta >:= \mathbb{Z}/p^\ell\mathbb{Z}[X]/f(X) \tag{1}$$

which is in particular equal to $\mathbb{Z}/p^\ell\mathbb{Z}$ when $r = 1$. This is a *free module* over $\mathbb{Z}/p^\ell\mathbb{Z}$ of dimention r. That is: it is isomorphic to $(\mathbb{Z}/p^\ell\mathbb{Z})^r$, with basis $1, \Delta, \ldots, \Delta^{r-1}$. Multiplication in $R_\ell(r)$ is defined by the multiplication modulo $f(X)$. Notice that an equivalent definition of $R_\ell(r)$ is to consider the unramified extension of degree r of the ring $\mathbb{Z}_p$ of p-adic integers, which is denoted $W(\mathbb{F}_{p^r})$ the "Witt ring", then reduce it modulo p^ℓ. This will be used in §3.3, and is also a useful point of view for the Hensel lifting algorithm of §4.

We say that an element $x \in R_\ell(r)$ is invertible modulo p if its reduction $\bar{x} \in \mathbb{F}_{p^r}$ is invertible. A key property of Galois rings is that an element invertible modulo p, is then also invertible in $R_\ell(r)$. Indeed, consider an arbitrary lift y of $\overline{x}^{-1}$. Then we have a formula $xy = 1 - p\lambda$ which holds in $R_\ell(r)$ for some λ. But the right hand side of the equation is invertible, of inverse $1 + p\lambda + \cdots + (p\lambda)^{\ell-1}$. From this formula we see that inversion in $R_\ell(r)$ costs essentially one inversion in $\mathbb{F}_{p^r}$, and $O(\log(\ell))$ squarings in $R_\ell(r)$.

2.1.2 Embeddings, and their equal complexities than over fields From the previous, we see that considering $\mathbb{Z}/p^\ell\mathbb{Z}$ as *embedded* in $R_\ell(r)$, multiplies by r the size (an element x is mapped to the vector $(x, 0, \ldots, 0)$ with r coordinates) by the same factor than when embedding $\mathbb{F}_p$ in $\mathbb{F}_{p^r}$. It follows from the definition (1) that the naive schoolboy multiplication algorithm in $R_\ell(r)$ has the same complexity than the one in $\mathbb{F}_{p^r}$. For large Galois rings, we have efficient multiplication algorithms, which are motivated by their usage in LWE. Hence, the references pointed in [Abs+19a, page 4] and [PC] show that they have also the same complexity than in $\mathbb{F}_{p^r}$.

Finally, one may also need to make the converse operation, and "descend" from secret sharing schemes over $R_\ell(r)$, to secret sharing schemes over $\mathbb{Z}/p^\ell\mathbb{Z}$.

The technique to do this over fields is introduced in [Cas+09, Theorem 7 & 8], and based on linear maps called "multiplication friendly embeddings (MFE)": $\mathbb{F}_{p^m} \longrightarrow \mathbb{F}_p^{2m-1}$, which have the property to bring the multiplication in $\mathbb{F}_{p^m}$, into the componentwise product in $\mathbb{F}_p^{2m-1}$. For the same reason as Reed Solomon codes lift trivially over rings, we have that the MFEs of [Cas+09, Theorem 8] carry over $\mathbb{Z}/p^\ell\mathbb{Z}$ with the same parameters, and thus we have exactly the same "expansion rates" $((2m-1)/m)$.

2.2 General LSSS & ASSSM over Rings

Let R be any finite ring (including $R = \mathbb{F}_{p^r}$ or $R_\ell(r)$), and n, k be positive integers. To share a secret s in R, one samples uniformly an element $\boldsymbol{w} \in R^{k-1}$ (the randomness space) then applies a certain linear map ψ on the whole to obtain n "shares" : $\psi(s, \boldsymbol{w}) \in R^n$. For $I \subset \{1, \dots n\}$ a set of indices, we denote $|I|$ the size of I and $\pi_I : R^n \to R^I$ the projection on these components. For any vector $\boldsymbol{x} \in R^n$, we denote for short $\boldsymbol{x}_I := \pi_I(\boldsymbol{x})$ this projection, i.e., the components of $\boldsymbol{x}$ in I, and likewise, for any linear map ψ in R^n, we denote for short $\psi_I := \pi_I \circ \psi$ the "components of ψ in I". Let $0 \leq t < n$ be a positive integer. Let $k, n \geq 1$ be integers, we say that a *linear secret sharing scheme* (LSSS) over R with n shares and randomness space R^{k-1}, is an R-linear map:

$$\begin{aligned} \psi : & R \times R^{k-1} \longrightarrow R^n \\ & (s, \boldsymbol{w}) \longrightarrow \psi(s, \boldsymbol{w}) \end{aligned}$$

We say that it has t-privacy if for any share vector, any t coordinates are independent of the secret, and it has *rec-reconstruction* if any rec coordinates of a vector of shares determine the secret s.

Definition 6. We say that a LSSS with privacy threshold t, is furthermore Arithmetic with Strong multiplication (ASSSM), if for any two secrets $s, s' \in R$, consider any sharings of them: $(s_i) = \psi(s, \boldsymbol{u})$ and $(s'_i) = \psi(s', \boldsymbol{u'})$, then for any set I of indices of size $n-t$, the data of the "sharewise" products $(\psi(s_i)\psi(s'_i))_{i \in I}$ determines uniquely ss'. Said otherwise, I is a "reconstruction set" for $\psi \times \psi$.

Notice that this a fortiori implies $n - t$ reconstruction threshold. If one replaces $n-t$ by n in the definition above, then this is the weaker *Multiplication* property.

2.3 Complexity of Sharing

From now on we specialize to a Galois ring $R := R_\ell(r)$ as defined in (1), e.g., equal to $\mathbb{Z}/p^\ell\mathbb{Z}$ when $r = 1$.

2.3.1 Proof of Th 5 (A): Systematic Form Let $C \subset R_\ell(r)^{n+1}$ be a *free* submodule of rank k, i.e., which is isomorphic to $R_\ell(r)^k$. Making a choice of $n+1$ coordinates in $R_\ell(r)^{n+1}$, we denote this a *"free code"*. Likewise, we say that k elements in $R_\ell(r)^{n+1}$ form a *free* family if they generate a submodule

isomorphic to $R_\ell(r)^k$ (we then say: "freely generate"). Recall that this implies that the reduction $\overline{C}$ of C modulo p is a vector space of *same dimension* k. [This follows immediately from the fact that if a square matrix with entries in $R_\ell(r)$ is invertible, then its reduction $\bmod p$ is invertible.] For the same reason, in the other direction, starting from a code $\overline{C}$ over $\mathbb{F}_{p^r}$ of dimension k, and considering any basis, then arbitrary lifts in $R_\ell(r)$ of these basis vectors generate a free code C of same rank k.

We denote that a matrix $G \in R_\ell(r)^{k\times(n+1)}$ is in *echelon form*, if for each row $i \in \{1,\ldots,k\}$, there exists a column $j_i \in \{0,\ldots,n\}$ containing a 1 entry on row i and 0 everywhere else. We say in particular that G is in *systematic form* if of the form $(\mathrm{Id}_k|N)$. We say that matrix G' is deduced from matrix G by "elementary row operations", if there exists a sequence of elementary row operations that transforms G into G'. Equivalently, if there exists an invertible matrix $E \in R_\ell(r)^{k\times k}$ such that $G' = EG$. Let us restate for convenience existence the systematic form of free codes, which is used at least since Calderbank-Sloane [CS95] (see also [SAS17, §5.1.1]). We re-prove it with an explicit construction, which has same complexity than over fields, which thus proves Theorem 5 (A).

Proposition 7. *Let $G \in R_\ell(r)^{k\times(n+1)}$ be a matrix such that the rows form a* free *family. Then there exists a matrix in echelon form which is obtained from G by elementary row operations. And thus, up to reordering the $n+1$, coordinates, in systematic form.*

Proof. Consider the reduction of $\overline{G}$ in $\mathbb{F}_{p^r}$. By the Gauss pivot, there exists an invertible $k \times k$ matrix $\overline{E}$ and a matrix $\overline{G'}$ in echelon form, such that $\overline{G'} = \overline{E}\overline{G}$. Let $E \in R_\ell(r)^{k\times k}$ be an arbitrarily lift of $\overline{E}$. E being invertible (since its determinant is invertible modulo p), the matrix $G' := EG$ is deductible from G by elementary row operations. G' being a lift of $\overline{G'}$, we have furthermore, for each row i, existence of a column j_i such that the entry G'_{i,j_i} is a lift of 1, and thus *invertible* in $R_\ell(r)$. Using this entry as a pivot, we anihilate all the other entries on this column j_i by elementary row operations. Finally, we divide the row i by G'_{i,j_i}, thus entry (i,j_i) becomes 1. Repeating for all i yields a matrix G'' deduced from G' by elementary row operations.

2.3.2 Sharing Up to permutation of the coordinates, we may now assume $G \in R_\ell(r)^{k\times(n+1)}$ of C in *systematic form*. By Th 5 (A) (Prop 7), the (one-shot) complexity of computing this form is essentially the same as over fields. Then, sharing a secret $s \in R_\ell(r)$ with respect to the 0-th coordinate of C, boils down to the following. First, sample a vector $\boldsymbol{w} \in R_\ell(r)^{k-1}$, uniformly at random. This has complexity $O(k)$ (or in terms of bits: $O(\ell k \log_2(p))$). Then, deduce the vector of shares from the left multiplication:

$$\psi : R_\ell(r) \times R_\ell(r)^{k-1} \longrightarrow R_\ell(r)^n \tag{2}$$

$$(s, \boldsymbol{w}) \longrightarrow (s, \boldsymbol{w})G_{[1,\ldots,n]}. \tag{3}$$

Where $G_{[1,\ldots,n]}$ denotes the n last columns of G. The complexity claim of Theorem 5(A') then follows from the fact that $\dim(\overline{C}) = \mathrm{rk}(C) = k$, and thus that

the generating matrices have the same sizes, combined with the fact pointed in §2.1, that complexity of the multiplication in $R_\ell(r)$ (by definition relatively to elementary operations in $\mathbb{Z}/p^\ell\mathbb{Z}$) is the same as the one in $\mathbb{F}_{p^r}$ (by definition relatively to $\mathbb{F}_p$).

2.4 Privacy (with Uniformity) and Efficient Reconstruction from *Free* Codes

Let us now bound privacy and reconstruction. The following states that Theorem 11.77 and Corollary 11.79 [CDN15] also hold over rings. Moreover, we will also prove computational efficiency of (4) (reconstruction) along the proof.

Proposition 8. *Let C be a* free *code in $R_\ell(r)^{n+1}$ of rank k. Denote $\overline{C}$ the code over $\mathbb{F}_{p^r}$ obtained by reduction modulo p, $\overline{C}^\perp$ the dual, and $d(\overline{C})$, $d(\overline{C}^\perp)$ the minimal distances. Consider the LSSS with n shares in $R_\ell(r)$ obtained from C. Recall that rec denotes the reconstruction threshold. Then we have:*

$$rec \leq n + 1 - (d(\overline{C}) - 1) = n - d(\overline{C}) + 2 \tag{4}$$

$$\text{For all} \quad t \geq d(\overline{C}^\perp) - 2 \quad \text{, we have that:} \tag{5}$$

each set of t shares is uniformly random in $R_\ell(r)^t$, in particular we have t-privacy.

2.4.1 Reconstruction: Constructive proof of (4), thus of Thm 5 (B)

Notice that Equation (4) is proven on a specific case in [Abs+20, Theorem 6]. But it actually holds in general. Let us take the opportunity to make a constructive proof, which will thus support our complexity claim of Thm 5 (B). We keep the notations of Equation (2).

Let $I \subset \{1, \ldots, n\}$ be a subset of $n + 1 - (d(\overline{C}) - 1)$ indices. By definition of the minimal distance, the linear map $\overline{\psi_I} : \mathbb{F}_{p^r} \times \mathbb{F}_{p^r}^{k-1} \longrightarrow \mathbb{F}_{p^r}^I$ is injective. Since it is defined over fields, it thus has a *linear left inverse*. We conclude by applying Lemma 9 to $M := R_\ell(r)^k$, $m := |I|$ and $f := \psi_I$.

Lemma 9. *Let M be a free $R_\ell(r)$-module (say of rank v) and $f : M \to R_\ell(r)^m$ be a R-linear map. Assume that the map modulo p:*

$$\overline{f} : (M \bmod p) = \mathbb{F}_{p^r}^v \longrightarrow \mathbb{F}_{p^r}^m$$

is an injection. Then f has a linear *left inverse $g : R_\ell(r)^m \longrightarrow M$. In particular, the image of f is a* free *$R_\ell(r)$-module.*

Proof. The matrix mat_f of f, of size $m \times v$ is such that, by assumption, when we reduce it modulo p, then it contains a $v \times v$ invertible minor. But then this minor in mat_f is also invertible (recall that $(1 + \lambda p)^{-1} = 1 + \sum_i \lambda^i p^i$). Inverting this minor (e.g. with Gauss pivot over $R_\ell(r)$, for efficiency), and completing with $m - v$ zero columns, yields a map $g : R_\ell(r)^m \longrightarrow R_\ell(r)^v$ such that $g \circ f = \mathrm{Id}_v$.

The last claim follow from the fact that f is in particular injective, so defines an isomorphism between M, which is free, and its image inside $R_\ell(r)^m$.

Our claim about the computational complexity then follows as previously from *linearity* of the reconstruction map ψ_I, and the fact (§2.1) that multiplications in $R_\ell(r)$ has same complexity than in $\mathbb{F}_{p^r}$.

2.4.2 Warning: Loss of Efficient Reconstruction for Non-Free Codes Recall that reconstructibility of a code means that, for any set of $d-1$ coordinates, the map consisting in puncturing these coordinates is an injection; and that *efficient* reconstruction means that it has a *linear left-inverse*, which we denote a *retraction*, as known as the *reconstruction map*. In the following Counterexample 10 we show that, without the assumption to be free, there exists submodules of $R_\ell(r)^{n+1}$ for which the puncturing map is an injection, but for which *there does not exists any linear retraction*. This motivates why we restrict to *free* codes in $R_\ell(r)^{n+1}$ in order to construct LSSS.

Counterexample 10. *Let C be a code in $R_\ell(r)^{n+1}$ with $\overline{d} := d(\overline{C}) \geq 2$ such that there exists a punctured $C^* \subset R_\ell(r)^{n+1-(\overline{d}-1)}$ which is* not free. *[For example*

$$C = \langle (p,p,p,0), (1,0,0,1) \rangle \in R_\ell(r)^4, \quad (\textit{e.g. } R_\ell(r) := \mathbb{Z}/p^\ell\mathbb{Z} \ )$$

with $d(\overline{C}) = 2$ and injectivity in $R_\ell(r)^3$ when puncturing the last coordinate.] Then there does not exist *any linear reconstruction map, i.e., any retraction $R_\ell(r)^{n+1-(\overline{d}-1)} \longrightarrow C$.*

The proof is that, supposing such a retraction, then, composing it on the left with the puncturing map, yields a left-inverse to the inclusion $C^* \subset R_\ell(r)^{n+1-(\overline{d}-1)}$. Denoting $G^* \in R_\ell(r)^{k \times (n+1-(\overline{d}-1))}$ a generating matrix of C^* (in rows) and $L \in R_\ell(r)^{(n+1-(\overline{d}-1)) \times k}$ the matrix of this left-inverse, we would thus have by assumption $G^*L = \mathrm{Id}_k$. In particular $\overline{G^*}$ modulo p would be of maximal dimension, k, thus G^* would generate a *free* module, a contradiction.

2.4.3 Privacy (with Uniformity): Proof of (5) The bound (5) is proven in [Abs+20, Theorem 6]: although on a specific LSSS, the arguments actually apply in general. The key to prove this formula over rings is their Lemma 3. Let us recall it here, and provide a both shorter and self-contained proof for it.

Lemma 11. *Let C be a submodule of $R_\ell(r)^n$, denote $\overline{d}^\perp$ the dual distance of the reduction $\overline{C}$ modulo p. Let $\mathcal{I}$ be a set of indices with $|\mathcal{I}| = \overline{d}^\perp - 1$. Then, projection of C on the indices in $\mathcal{I}$ is the full space $R_\ell(r)^{\mathcal{I}}$.*

Proof. By assumption, $\overline{C}_{\mathcal{I}} = \mathbb{F}_q^{\mathcal{I}}$. Hence $C_{\mathcal{I}}$ contains $|\mathcal{I}|$ vectors so that the matrix formed by them has an invertible determinant, thus has an inverse, thus these vectors generate $R_\ell(r)^{\mathcal{I}}$.

Then, the bound (5) follows by applying the Lemma to any set $\mathcal{I}_{\mathcal{A}}$ of t indices, to which we add the index $\{0\}$. Indeed, we then have surjectivity of the

projection $C \longrightarrow C_{\{0\}} \times C_{\mathcal{I}_\mathcal{A}}$. In particular, for any fixed secret $s \in R_\ell(r)$ (0^{th} coordinate), we have surjectivity of the projection from the affine submodule C_s of codewords with 0^{th} coordinate equal to s, onto any subset of t shares. Thus by definition the shares of s under the LSSS are such that any t of them vary uniformly in $R_\ell(r)^t$, which was to be proven.

2.5 (Free) Generation from Any Lift of *Any* Basis

The following important fact is not formalized in the literature to our knowledge:

Theorem 12. *Let C be a* free *code in $R_\ell(r)^{n+1}$. Consider the reduced code modulo p: $\overline{C} \subset \mathbb{F}_{p^r}$, and* any *basis $(\overline{e'}_i)$ of $\overline{C}$. Then C is freely* generated *by* any *lift of $(\overline{e'}_i)$ inside C.*

Proof. Let k denote the rank of C. The "freely" claim again follows from the fact that a family whose reduction modulo p is free, is itself free (the generating matrix containing an invertible $k \times k$ determinant). Now, consider $(e'_{i=1\dots k})$ an arbitrary lift of the $(\overline{e'}_i)$ inside C. It generates a submodule in C, which is free of rank k by the first part of the proof. But C is itself a free module of rank k. Thus this defines an injection $R_\ell(r)^k \hookrightarrow R_\ell(r)^k$, which is by assumption also an injection modulo k. Thus by Lemma 9 it has a left inverse, thus it is a bijection.

Corollary 13. *If $E \subset G$ are two lifts R^n of the same code $\overline{G}$, and G is free, then they are equal (in particular E is also free).*

Proof. Indeed E contains a lift of a basis of $\overline{G}$ which, by Theorem 12, generate the whole G. □

3 Main Theorem 1

3.1 A Random Free Lift of a Code of *Small Square* mostly *Fails* to Have a Small Square

For C a code (over a field or a ring), we denote as *componentwise square* C^{*2} the code of same length which is generated by all the products of any two codewords of C component by component. Strong multiplication of the LSSS from C thus requires that C^{*2} has large distance, thus be of small size. The central problem of this paper is, starting from a free code $C \in R_\ell(r)^n$ (typically $\ell = 1$, i.e., $R_\ell(r)$ is a finite field) which has free square of small dimension, then find a code $\widetilde{C}$ in a larger ring than C, that reduces to $C \bmod p^\ell$, is also free, and has square of small size. Ideally, the square $\widetilde{C}^{*2}$ is desired to be also *free*, in which case it is automatically of same rank as C^{*2} (since the determinant is invertible mod p^ℓ). We denote this desirable object informally as a "multiplication friendly lift" in the exposition, whereas in the statements it will be replaced by precise specifications. Let us revisit the family of [Abs+20, Example 2], and explain why they provide also counterexamples where arbitrarily lifting *fails* to yield a multiplication friendly lift.

Counterexample 14. *Let $\bar{C}$ and $\bar{D}$ be codes over $\mathbb{F}_{p^r}$ of same dimension and let us assume that $\dim \bar{D}^{*2} < \dim \bar{C}^{*2}$. Let us now build a code E over $R_\ell(r)$ with $\ell \geq 3$ and of length equal to the sum of the lengths of $\bar{C}$ and $\bar{D}$. Let $(\bar{c_i})_i$ and $(\bar{d_i})_i$ be bases of $\bar{C}$ and $\bar{D}$, let $(c_i)_i$ and $(d_i)_i$ be arbitrary lifts and define E the code generated by the vectors $(d_i, pc_i)_i$. Then E is free, because of dimension $\dim \bar{D} = \dim \bar{E}$, and is a lift of $\bar{E}$. Suppose by contradiction that the square E^{*2} would be free, then we would have:*

$$\dim E^{*2} \geq \dim \bar{C}^{*2} > \dim \bar{D}^{*2} = \dim \bar{E}^{*2} \ .$$

On the other hand if it was free, then it would be of same rank than $\bar{E}^{*2}$ by Theorem 12. So we have a contradiction. Thus E^{*2} is not free, thus it is strictly larger than some free lift of $\bar{E}^{*2}$ inside him.

3.1.1 The desirable case of small square: sparsity of solutions, if any, illustrated on a toy example Let us now illustrate hardness of the multiplicative lifting problem on a tiny AG code. Consider the elliptic curve $y^2 + xy + y - x^3 + 1$ over

$$\mathbb{F}_{2^3} = \mathbb{F}_2 < \delta > \text{ with polynomial } \delta^3 + \delta + 1 = 0,$$

with 14 places, P_0 the place at infinity, the divisor $D_0 = 4P_0$ and the Riemann-Roch space $L(4P_0)$, with basis $\overline{\boldsymbol{e_i}}_{i=1\ldots4}$ equal to the functions $(1, x, x^2, y)$. Let us define the evaluation code $C(D_0)$ at the $P_1, \ldots, P_{13}$, (not at P_0, for simplicity). We compute the following generating matrix:

$$\overline{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \delta & \delta & \delta^2 & \delta^2 & \delta^3 & \delta^3 & \delta^4 & \delta^4 & \delta^5 & \delta^5 & \delta^6 & \delta^6 & 1 \\ \delta^2 & \delta^2 & \delta^4 & \delta^4 & \delta^6 & \delta^6 & \delta & \delta & \delta^3 & \delta^3 & \delta^5 & \delta^5 & 1 \\ 1 & \delta & 1 & \delta^2 & \delta^2 & \delta^4 & 1 & \delta^4 & \delta & \delta^2 & \delta & \delta^4 & 0 \end{bmatrix}$$

Let us consider the 10 componentwise products $\overline{\boldsymbol{e_i}} * \overline{\boldsymbol{e_j}}$, with indices (i, j) ordered as: $(1,1), (1,2), (1,3), (1,4), (2,2), (2,3)$ etc. (i.e.: j increases first). They generate by definition $C(D_0)^{*2}$. We verify that, removing $(2,2)$ and $(4,4)$ from the indices in this list, then the remaining 8 products: $\overline{\mathcal{B}} := (\overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}})_{(k,l)\in B}$ generate $C(D_0)^{*2}$, where B denotes the remaining indices ordered as before. In particular $\overline{\boldsymbol{e_2}} * \overline{\boldsymbol{e_2}}$ and $\overline{\boldsymbol{e_4}} * \overline{\boldsymbol{e_4}}$ decompose themselves on this basis $\overline{\mathcal{B}}$, with decomposition coefficients $(\overline{\lambda_{2,2,k,l}})_{k,l\in B}$ and $(\overline{\lambda_{4,4,k,l}})_{k,l\in B}$ given by the following 2×8 matrix, called "Reduc" in the implementation:

$$\left(transp(\overline{\lambda_{2,2,k,l}}, \overline{\lambda_{4,4,k,l}})\right)_{(k,l)\in B} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{6}$$

Then we repeated the following experiment 10^8 times: *randomly* lift the $(\overline{\boldsymbol{e_i}})_i$ modulo 2^2, to obtain vectors $(\boldsymbol{e_i})_i$ with coordinates in $R_2(3) = \mathbb{Z}/2^2\mathbb{Z} < \Delta >$. Let C_{bad} the code generated by these lifts. By Theorem 12, it is always *free*. But we observed in *all the experiments* that $\boldsymbol{e_2} * \boldsymbol{e_2}$ and $\boldsymbol{e_4} * \boldsymbol{e_4}$ *do not* anymore

decompose themselves on the lifts of the previous basis of $C(D_0)^{*2}$: $\mathcal{B} := (\boldsymbol{e_k} * \boldsymbol{e_l})_{(k,l)\in B}$ —see two paragraphs later for an explanation of how this checks were done efficiently with linear algebra. So in these situations C_{bad}^{*2} *is not* a free lift of the square $C(D_0)^{*2}$, because if it were, then by Theorem 12 the lifted basis $\mathcal{B}$ *would* generate it.

3.1.2 Why solutions may likely not exist at all Let us give a feeling of why most codes with small squares are likely to have no multiplication friendly lift. Let $\overline{C}$ be a code over, say, $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ of dimension k and length n, such that the square $\overline{C}^2$ has *small* dimension, say, $3k < n$. We would like to find a code C over $\mathbb{Z}/p^2\mathbb{Z}$ (namely: a free submodule of $(\mathbb{Z}/p^2\mathbb{Z})^n$) of same rank k, that lifts $\overline{C}$ modulo p^2, and such that the square C^2 is also a free lift of $\overline{C}^2$. As argued with the toy example, it follows from Theorem 12 that these requirements are equivalent to the following: let $(\boldsymbol{e_i})_i$ be any basis of C lifting a basis $(\overline{\boldsymbol{e_i}})$ of $\overline{C}$; let $\overline{\mathcal{B}}$ be *any* basis of $\overline{C^2}$; then $\overline{\mathcal{B}}$ lifts modulo p^2 to a basis of the square C^2, in particular generates the componentwise products $(\boldsymbol{e_i} * \boldsymbol{e_j})_{i,j}$. To fix ideas let us choose a basis of the form $\overline{\mathcal{B}} = (\overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}})_{(k,l)\in B}$ as in the toy example. Then the previous equivalent condition translates itself into the fact that the equations expressing $\overline{\boldsymbol{e_i}} * \overline{\boldsymbol{e_j}}$ on this basis:

$$\overline{\boldsymbol{e_i}} * \overline{\boldsymbol{e_j}} = \sum_{(k,l)\in B} \overline{\lambda_{i,j,k,l}}\, \overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}} \pmod{p} \tag{7}$$

lift modulo p^2. The number of degrees of freedom (the unknowns) are: (i) the choices of lifts for the $\overline{\boldsymbol{e_i}}$, so a total of nk coordinates to lift in $\mathbb{Z}/p^2\mathbb{Z}$; (ii) and lifts for the coefficients $\overline{\lambda_{i,j,k,l}}$: a total of $3k \times k(k+1)/2$ unknowns in $\mathbb{Z}/p^2\mathbb{Z}$. So the number of unknowns is asymptotically equivalent to (ii): $3k\times k(k+1)/2$. Whereas the number of equations is $nk(k+1)/2$ (namely: $k(k+1)/2$ vectorial equations with n coordinates in $\mathbb{Z}/p^2\mathbb{Z}$ each). Notice that $3k < n$, so that there are more constraints than variables. Finally, as will be detailed in the next paragraph, and then further in §4 notice that this *quadratic* system over a *ring* simplifies modulo p^2 to a *linear system* over the *field* $\mathbb{F}_p$. Thus, the system being *overdetermined*, then *a priori no solution is likely to exist.*

3.2 A technique to find them when they exist, illustrated on the toy example

We will formalize the general technique in §4. Existence of a solution to the system for AG codes, is further evidence that these codes are *highly non-generic* among those with small square.

First, fix a free lift C_{bad} of $C(D_0)$ by lifting arbitrarily the basis to $(\boldsymbol{e_i}')_i$, for example by lifting the coordinates from $\mathbb{F}_2 < \delta >$ to $\mathbb{Z}/2^2\mathbb{Z} < \Delta >$ by the dummy rule: $1 \to 1$ and $\delta \to \Delta$. This gives formally the same generating matrix as $\overline{G}$, with δ replaced by Δ. With the same dummy rule, lift the decomposition coefficients $(\overline{\lambda_{2,2,k,l}})_{(k,l)\in B}$ and $(\overline{\lambda_{4,4,k,l}})_{(k,l)\in B}$ to $\lambda'_{2,2,k,l}$ and $\lambda'_{4,4,k,l}$, so that

their matrix is formally the same as in (6). As the case for the huge majority of arbitrarily chosen lifts (and illustrated with random tests two paragraphs above), the vectors $\boldsymbol{e'_2} * \boldsymbol{e'_2}$ and $\boldsymbol{e'_4} * \boldsymbol{e'_4}$ *do not* decompose themselves on $\mathcal{B} := (\boldsymbol{e_k}' * \boldsymbol{e_l}')_{(k,l)\in B}$, let alone with coefficients equal to $\lambda'_{2,2,k,l}$ and $\lambda'_{4,4,k,l}$. As a matter of fact, we encounter nonzero error vectors $2\boldsymbol{D_{2,2}}$ and $2\boldsymbol{D_{4,4}}$ when trying to write the decompositions in $\mathbb{Z}/2^2\mathbb{Z} < \Delta >$:

$$(8) \qquad \boldsymbol{e'_2} * \boldsymbol{e'_2} = \sum_{(k,l)\in B} \lambda'_{2,2,k,l} \boldsymbol{e_k}' * \boldsymbol{e_l}' + 2\boldsymbol{D_{2,2}} \text{ and likewise for } \boldsymbol{e'_4} * \boldsymbol{e'_4}$$

Let us insist on the remarkable fact that the error vectors are multiples of 2, since the equalities (8) do hold without error term modulo 2. "Dividing" by 2, their coefficients are

$$transp(\boldsymbol{D_{2,2}}, \boldsymbol{D_{4,4}}) = \begin{bmatrix} 0 & 0 & \delta^4 & \delta^4 & \delta & \delta & 1 & 1 & \delta^5 & \delta^5 & \delta & \delta & 0 \\ 0 & 0 & \delta & \delta & 0 & 1 & \delta^2 & \delta^2 & \delta^4 & 0 & 0 & \delta^5 & 1 \end{bmatrix}$$

Which we express in $\mathbb{F}_{2^3}$ by abuse of notation (remember that an element $2x \in \mathbb{Z}/2^2\mathbb{Z} < \Delta >$ is determined by the residue $\overline{x} \in \mathbb{F}_{2^3} \mod 2$). Now, let us look for corrective terms $2\boldsymbol{f'_i}$ and $2\mu'_{i,j,k,l}$, which we need only to find modulo 2:

$$(9) \qquad \boldsymbol{e_i} = \boldsymbol{e'_i} + 2\boldsymbol{f'_i} \text{ and } \lambda_{i,j,k,l} = \lambda'_{i,j,k,l} + 2\mu'_{i,j,k,l}$$

So that, replacing $\boldsymbol{e_i}'$ in (8) by the corrected $\boldsymbol{e_i}$ of (9) —where the corrective terms are treated as unknows—, simplifying and removing the terms that are multiples of 2^2 —because they vanish in $\mathbb{Z}/2^2\mathbb{Z} < \Delta >$—, we observe that all the terms remaining in the system are multiples of 2. So "dividing" the system by 2, we fall back to a *linear* system in $\mathbb{F}_{2^3}$:

(10)

$$\overline{\boldsymbol{e_2}} * \boldsymbol{f'_2} + \overline{\boldsymbol{e_2}} * \boldsymbol{f'_2} - \boldsymbol{D_{2,2}} = \sum_{(k,l)\in B} \mu'_{2,2,k,l} \overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}} + \overline{\lambda_{2,2,k,l}} (\overline{\boldsymbol{e_k}} * \boldsymbol{f'_l} + \overline{\boldsymbol{e_l}} * \boldsymbol{f'_k}) \quad (\text{same for } \overline{\boldsymbol{e_4}} * \overline{\boldsymbol{e_4}})$$

as could be expected from Hensel's Lemma. Solving this system for the corrective terms, we deduce the corrected basis $(\boldsymbol{e_i})_i$ defined as in (9), that define the corrected code C_{good}, whose coordinates are given in the big left-hand rotated matrix on the first formula page of the Appendix.

Likewise we deduce the corrected decomposition coefficients $(\lambda_{2,2,k,l})_{k,l\in B}$ and $(\lambda_{4,4,k,l})_{k,l\in B}$ as given in the centered right-hand formula.

We can finally check straightforwardly that, with these corrected values, then $\boldsymbol{e_2} * \boldsymbol{e_2}$ and $\boldsymbol{e_4} * \boldsymbol{e_4}$ now decompose themselves on $\mathcal{B}$ with the corrected coefficients, without anymore parasitic error vectors. So with these corrected lifts $(\boldsymbol{e_i})_i$, we have now that the square of the corrected code C_{good} is also a free lift. That is, we have succeeded in modifying the free lift C_{bad} into a *multiplication-friendly* lift C_{good}.

3.3 Proof of Main Theorem 1

3.3.1 Roadmap of the proof.

First Consider a smooth curve over $\mathbb{F}_{p^r}$ and a divisor D_0 on this curve (that is: a set of points with multiplicities), such that the degree (the sum of the multiplicities) is $\deg(D_0) < n$. Then, the curve has a lift defined over the ring $R_\ell(r)$ (provided it is given under an equivalent form where equations have no singular points). Lifting the points then applying Judy Walker's results, we have the existence of lifts of the Riemann-Roch spaces: $L(D)$ and $L(2D)$ which are *free* modules, and such that we have inclusions of products of spaces of global sections

$$L(D)^{\otimes 2} = L(2D) , \tag{11}$$

where the traditional notation $L(D)^{\otimes 2}$ stands for the space generated by all products fg of pairs of sections (f, g) in $L(D)$. Then, from Judy Walker's Theorem 15 below, we deduce that the evaluation codes over rings $C(D)$ and $C(2D)$, arising from evaluation of these free lifts of Riemann-Roch spaces, are also *free*. We will detail this material in the two next subsections

Next The key property of these free lifts is that they behave well with respect to inclusions and squares:

$$C(D)^{*2} \subset C(2D) . \tag{12}$$

Here the code $C(2D)$ is free for the same reasons, with same rank as the classical AG code $\overline{C}(2D_0)$ below modulo p. So this forces the square $C(D)^{*2}$ to stay small, contrary to the square of an arbitrary free lift, which may "spread out" too much (as seen in Counterexample 14).

Deducing the parameters By freeness of $C(D)$, Proposition 8 (5) implies that a LSSS from $C(D)$ has privacy threshold at least as large as a LSSS from the code below modulo p: $\overline{C}(D_0)$.

Likewise, by Proposition 8 (4), a LSSS from the *free* code $C(2D)$ has *full reconstruction* from any $n - d(\overline{C}(2D_0)) + 2$ shares. Thus, by inclusion (12), so does a LSSS from the subcode $C(D)^{*2}$. Said otherwise, a LSSS from $C(D)$ has *reconstruction of the product*, from a number of pairwise products of shares which is *as small* as for a LSSS from $\overline{C}(D_0)$.

For sake of completeness we review the concrete parameters of these schemes in §3.4, examplified on the ones of [CC06].

3.3.2 Lift of curves, divisors and Riemann-Roch spaces Let us follow Walker's [Wal99] notations. Note $R = R_\ell(r)$ the (Artinian local) Galois ring, with residue ring $R/(p) = \mathbb{F}_{p^r}$. X_0 being a smooth projective curve over $\mathbb{F}_{p^r}$, then from [Ill05, Theorem 5.19 ii)] (or [SGA1, III Corollaire 7.4]), X_0 has a smooth projective lift over the ring of Witt vectors $W(\mathbb{F}_{p^r})$. Which, after reduction mod p^ℓ, yields a projective lift X over R (because these properties are preserved by base change). Also, R being local, $\mathbb{F}_{p^r}$-points of X_0 lift to R-points of X by the formal smoothness criterion (see [Wal99, Remark 4.5] or next paragraph for details). As a consequence, divisors with support on rational points (actually any divisor) lift to X —and thus also do the line bundles $\mathcal{L}_0$ arising from them.

An explicit procedure for simultaneous compatible free lifts of line bundles. By [Wal99, Lemma 4.4] we can construct lifts of divisors D_0 on X from the following recipe. First, for every rational point $P_0^{(j)}$ of X_0, fix a closed point of degree one $P^{(j)}$ of X above P_0, as described in [Wal99, Remark 4.5] (lift arbitrarily $P_0^{(j)}$ to an R-point, then choose $P^{(j)}$ inside the image).

Then we can simultaneously lift divisors D_0 and $2D_0$ on X_0 as follows. For every rational point P_0 of X_0 in the support of the line bundle D_0, let m be the valuation of D_0 at P_0 and let P be the closed point lying above P_0 as fixed earlier. Deduce from it a divisor mP, then sum over the points P_0 in the support of D_0, to obtain a lift D of D_0. Likewise for the divisor $2D$, equal to the same formal sum of R-points as in D and with twice the multiplicities. In particular, note $\mathcal{L} := \mathcal{L}(D)$ the line bundle associated to D, and likewise for $\mathcal{L}(2D)$.

Proof of (11) This formula well known for curves over fields. Let us justify that it also holds over rings. The reason is that, by smoothness of the lift of the curve, this guarantees that, in a small enough neighborhood U of P, we also have a *uniformizer* denoted t_U (see [Wal, Proposition 4.9]). Thus, as long as U does not contain the other points of the support of D, we have:

$$\mathcal{L}_U = t_U^{-m}\mathcal{O}_U \ . \tag{13}$$

Thus $t_U^{-m}t_U^{-m} \in \mathcal{L}_U(2D)$, hence the claimed inclusion of products of global sections (11).

3.3.3 Deducing AG codes by Evaluation of Global Sections. For any divisor D on X, we denote as the "Riemann-Roch space" $\Gamma(X, \mathcal{L})$ the space of global sections. In the rest of the paper it is denoted instead $L(D)$. By the argument above [Wal99, Theorem 4.7], f is a *free* R-module that reduces modulo p to $\Gamma(X_0, \mathcal{L}_0)$. With slightly narrower conditions on the degree, then have the following compatibilities, as wrapped-up in [Wal99, Theorem 5.5]:

Theorem 15 (Lifts of Riemann-Roch spaces and AG codes). *Consider n rational points $\mathcal{P}_0 = \left(P_0^{(j)}\right)_{j=1\ldots n}$ on X_0, D_0 a divisor of degree:*

$$2g - 2 < \deg D_0 < n$$

with associated line bundle $\mathcal{L}_0$, and the injective evaluation map γ_0 yielding an algebraic geometry code $\overline{C}$ in $\mathbb{F}_{p^r}^n$. Then this data lifts to objects over R: X,$\mathcal{P}$ and D, with associated line bundle $\mathcal{L}$, yielding an evaluation code C, such that

we have the following commutative diagram:

$$
\begin{array}{ccccc}
\Gamma(X,\mathcal{L}) & \twoheadrightarrow & \Gamma(X,\mathcal{L})\otimes_R \mathbb{F}_{p^r} & \xrightarrow{\sim} & \Gamma(X_0,\mathcal{L}_0) \\
\Big\downarrow{\scriptstyle eval} & & & & \Big\downarrow{\scriptstyle eval} \\
\oplus_j \Gamma(P^{(j)},\mathcal{L}|_{P^{(j)}}) & & & & \oplus_j \Gamma(P_0^{(j)},\mathcal{L}_0|_{P_0^{(j)}}) \\
\Big\downarrow{\scriptstyle \cong}\gamma & & & & \Big\downarrow{\scriptstyle \cong}\gamma_0 \\
R^n & & \xrightarrow{\bullet\otimes_R \mathbb{F}_{p^r}} & & \mathbb{F}_{p^r}^n
\end{array}
\tag{14}
$$

Where: - the top left horizontal arrow and the bottom horizontal arrow are tensorisation by $\otimes_R \mathbb{F}_{p^r}$ - the top right isomorphism is constructed canonically as in the proofs of [Wal99, Lemma 4.6 & proof of Th 4.7]

- the top vertical arrows are the canonical restriction maps - the bottom left vertical arrow is a collection of arbitrary isomorphisms for all j:

$$\gamma_j : \Gamma(P^{(j)},\mathcal{L}|_{P^{(j)}}) \longrightarrow A$$

that reduce to γ_0 by tensorisation by $\otimes_R \mathbb{F}_{p^r}$ (and if not, then redefine γ_0 accordingly without changing the code in $\mathbb{F}_{p^r}^n$).

Notice that the name "evaluation maps" of the top vertical arrows is abusive in general (because of poles, etc: see the first example of §4.1), but they do play this role.

In conclusion, as explained in [Wal99], the code $C(D)$ (likewise $C(2D)$) is free because it is the image of a free module: $\Gamma(X,\mathcal{L})$, under the evaluation map which is an injection modulo (p), and thus its image is a *free* submodule of $R_\ell(r)^n$ by Lemma 9.

3.4 Reminders on the asymptotic parameters

Recall first the tradeoff of [CC06, §5] for secret sharing in finite fields $\mathbb{F}_p$. Let us cast a secret in $\mathbb{F}_p$, into the extension $\mathbb{F}_{p^r}$ of degree r, such that

$$p^r \geq 49 \ .$$

Then for adversary threshold $1/3-\epsilon$, and for infinitely many number of players, there exists an ASSSM over $\mathbb{F}_{p^r}$ and size r of shares, such that:

$$\epsilon < \frac{4}{3(p^{r/2}-1)} \ ,$$

[CC06, §5] In particular, choosing $\widehat{r}(\epsilon) = -2\log(\epsilon)$ yields an adversary bound $1/3-\epsilon$ when ϵ is sufficiently small.

Notice that the classical bound for the dual distance of AG codes over fields is not stated explicitly in [Cas+09; CCX11]. But its parameters are well known since Goppa (recalled e.g. in [Wal99, Theorem 2.1]), and also asymptocially optimal in our regime $2g-2 < \deg D_0 < n$. Which supports the claims of [Cas+09; CCX11], and thus ours by Proposition 8.

4 Computing Hensel Lift of a Code with a Small Square

Starting from any code $\overline{C} \subset \mathbb{F}_{p^r}$, for any positive L, the following Hensel lift algorithm lifts the code to a free code $C_L \subset R_L(r)^n$ such that the square remains generated by a lift of a basis of the square $\overline{C}^{*2}$. It proceeds in L recursive steps. Each of the steps consists in solving one instance of the same linear system over $\mathbb{F}_{p^r}$, of size $O(n^3) \times O(n^3)$. Thus the overall complexity is linear in L and polynomial in n. The algorithm was already illustrated in §3.2, let us formalize it.

Let $(\overline{\boldsymbol{e_i}})_{i\in[\dim \overline{C}]}$ be a basis of the code $C_1 := \overline{C}$. By definition, the square $\overline{C}^{*2}$ is generated by the $n(n+1)/2$ distinct componentwise products $(\overline{\boldsymbol{e_i}} * \overline{\boldsymbol{e_j}})_{(i,j)}$. $\overline{C}_1^{*2}$ being a vector space, one can extract a basis from the previous family, which we denote $(\overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}})_{(k,l)\in B}$, where $\left|B\right| = \dim \overline{C}_1^{*2}$.

A recursive step is as follows. The input is a free lift $C_\ell \in R_\ell(r)^n$ of $\overline{C}$, together with a basis $(\boldsymbol{e_i})_{i\in[\dim \overline{C}]}$, and coefficients $\left(\lambda_{i,j,k,l}\right)_{i\leq j,\,(k,l)\in B}$ in $R_\ell(r)$, such that we have the following invariant. The family of componentwise products $(\boldsymbol{e_k} * \boldsymbol{e_l})_{(k,l)\in B}$ *generates* the square C_ℓ^{*2}. The coefficients express the larger generating family $(\boldsymbol{e_i} * \boldsymbol{e_j})_{(i\leq j)}$ on the smaller generating family, namely:

$$\boldsymbol{e_i} * \boldsymbol{e_j} = \sum_{(k,l)\in B} \lambda_{i,j,k,l}\ \boldsymbol{e_k} * \boldsymbol{e_l} \quad \text{for all } i \leq j \tag{15}$$

The output of a step is a lift $C_{\ell+1} \in R_{\ell+1}(r)^n$, together with a basis $(\boldsymbol{e_i}'')_{i\in[\dim \overline{C}]}$ that lifts $(\boldsymbol{e_i})_{i\in[\dim \overline{C}]}$, and coefficients $\left(\lambda''_{i,j,k,l}\right)_{i\leq j,\,(k,l)\in B}$ in $R_{\ell+1}(r)$ that lift the $\left(\lambda_{i,j,k,l}\right)_{i\leq j,\,(k,l)\in B}$ such that the same invariant holds (this time with respect to the square $C_{\ell+1}^{*2}$).

The computation of a step is as follows. Fix *arbitrary* lifts $\boldsymbol{e_i}'$ of the $\boldsymbol{e_i}$ in $R_{\ell+1}(r)^n$, and $\lambda'_{i,j,k,l}$ of the $\lambda_{i,j,k,l}$ in $R_{\ell+1}(r)$. We obtain error terms $p^\ell D_{i,j}$ when evaluating the equations in $R_{\ell+1}(r)^n$:

$$\boldsymbol{e_i}' * \boldsymbol{e_j}' = \sum_{k,l} \lambda'_{i,j,k,l} \boldsymbol{e_k}' * \boldsymbol{e_l}' + p^\ell D_{i,j} \text{ for all } i \leq j \tag{16}$$

Solving the system means finding correct lifts $\boldsymbol{e_i}''$ and $\lambda''_{i,j,k,l}$ such that the error terms $p^\ell D_{i,j}$ are all equal to 0. We express $\boldsymbol{e_i}''$ and $\lambda''_{i,j,k,l}$ from $\boldsymbol{e_i}'$ and $\lambda'_{i,j,k,l}$, added with corrective terms $p^\ell \boldsymbol{f_i'}$ and $p^\ell \mu'_{i,j,k,l}$:

$$\boldsymbol{e_i}'' = \boldsymbol{e_i}' + p^\ell \boldsymbol{f_i'} \text{ and } \lambda''_{i,j,k,l} = \lambda'_{i,j,k,l} + p^\ell \mu'_{i,j,k,l} \tag{17}$$

So that, replacing $\boldsymbol{e_i}'$ in (16) by the corrected $\boldsymbol{e_i}''$ of (17) (where the corrective terms are treated as unknows), simplifying and moding out the terms that are multiples of $p^{\ell+1}$, we observe (Hensel's trick) that all the terms remaining in the

system are multiples of p^ℓ. Thus, dividing by p^ℓ, we fall back to the following *linear* system in $\mathbb{F}_{p^r}$:

$$(18)\quad \overline{\boldsymbol{e_i}} * \boldsymbol{f'_j} + \overline{\boldsymbol{e_j}} * \boldsymbol{f'_i} - D_{i,j} = \sum_{k,l} \mu'_{i,j,k,l} \overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}} + \overline{\lambda_{i,j,k,l}} (\overline{\boldsymbol{e_k}} * \boldsymbol{f'_l} + \overline{\boldsymbol{e_l}} * \boldsymbol{f'_k}) \ \forall i \leq j$$

which we notice is the *same* system for all steps. Finally, as for the size of the system, each vectorial equation for (i,j) expands itself in n scalar equations, so a total of $nk(k+1)/2$. The lifts of the $(\boldsymbol{e_i})_i$ are n unknowns and the lifts of $\lambda_{i,j,k,l}$ are $k(k+1).\dim(\overline{C}^{*2})$ unknowns.

Complexity in $\log(L)$ It was suggested by a reviewer of Eurocrypt that, applying the Hensel lifting method in its full version would enable a lifting complexity in only $O(\log_2 L)$ steps. This comes from the possibility to lift (15) directly modulo $p^{2\ell}$ (full Hensel method). However, this requires to determine the corrective terms modulo p^ℓ, and not anymore just modulo p as in (18). This thus requires the task of solving a linear system modulo p^ℓ, not anymore just modulo p. This task is efficiently computable, as proven in §5 (A). But for simplicity, we nevertheless implemented the method in L steps.

4.1 Example of a multiplication friendly lift modulo 2^{100}

Here we illustrate efficiency of our method by lifting a strongly multiplicative secret sharing scheme over $\mathbb{F}_{16}$ for 64 players and adversary threshold $t = 13$, into a scheme over $\mathbb{Z}/2^{100}\mathbb{Z}$, in a minute on a single processor.

Let X_0 be the "Hermitian" plane curve over $\mathbb{F}_{16}$ defined by equation $f(x,T) = T^4+T-x^{4+1}$. Then it is well known that this curve has genus $g = 4(4-1)/2 = 6$ and $n+1 := |X_0(\mathbb{F}_{16})| = 1+4^3 = 65$ rational points (which reaches the Hasse-Weil upper-bound). Let us denote these points $P_0, \ldots, P_{n=64}$, consider the divisor $D_0 = 25P_0$, whose Riemann-Roch space $L(D_0)$ is of dimension 20. Let $\overline{C}$ be the algebraic geometry code $\overline{C}$ of length $n+1$ defined as evaluations of $L(D_0)$ on all the rational points of X_0, *including the support* $\{P_0\}$ of D_0. Phrased with the notations of [CC06, §3], this means that we allow in addition to evaluate at Q. We do this to enable $+1$ on the adversary bound t. Evaluate at a point P_0 of the support of D_0, simply proceeds by pre-multiplying the function to be evaluated, by a uniformizer of P_0 to the power the order of P_0 in D_0. For the sake of illustration notice that, with $t = 13$, we have $\deg D_0 = 2g + t$ so that the condition $39 = 3t < n - 4g = 40$ of [CC06, Proposition 2] is satisfied, thus from $\overline{C}$ we can deduce a secret sharing scheme with strong multiplication for adversary bound $t = 13$.

Before going on, we compute the square code $\overline{C}^2$ and the (a priori larger) AG code associated to $L(2D_0)$, and check that both are equal. By the Riemann-Roch formula we have that $\overline{C(2D_0)}$ is of dimension $2.25 + 1 - 6 = 45$. From the generating set $(\overline{\boldsymbol{e_i}} * \overline{\boldsymbol{e_j}})_{i \leq j}$ of $\overline{C}^2$ we extract a basis $(\overline{\boldsymbol{e_k}} * \overline{\boldsymbol{e_l}})_{(k,l) \in B}$. We now look at the matrix expressing the $(\overline{\boldsymbol{e_i}} * \overline{\boldsymbol{e_j}})_{i \leq j}$ in terms of this basis (with the previous notations, this is the matrix of the coefficients $\overline{\lambda_{i,j,k,l}}$). It has $(\dim(C)(\dim(C)+$

1))/2 = 210 lines (all ordered pairs $i \leq j$). Obviously the lines where the index (i, j) belongs to B contain a single coefficient, equal to one. And obviously these coefficients will remain equal to one in every lift mod p^ℓ so we can remove these $n \dim \overline{C}^2 = 64 \times 45$ relations (and the corresponding variables) from the system from now on. This means that equality actually holds in (12). It is left outside of the scope of the paper to prove why this equality is actually implied by the condition $\deg D_0 \geq 2g + 1$.

After some optimizations outside of the scope of this paper, we end up with a system (18) of 10725 equations with 3305 unknowns but, surprisingly, of (still) very large kernel: dimension 83 (dimension 200 before applying the trick). We solve it in one second on a single processor.

Finally we repeat the operation, following the Hensel-lift algorithm: we reinject the solution (the lifted vectors $\boldsymbol{e_i}$ and coefficients $\lambda_{i,j,k,l}$) in a system mod 2^3 (as in (15)), which is a multiple of 2^2 after simplification, thus falls back to a system mod 2 after "division by 2^2". Note the general fact that the matrix of the new system obtained is exactly the same as the initial one (18), because the coefficients depend only on the values modulo 2 of $\overline{\boldsymbol{e_i}}$ and $\overline{\lambda_{i,j,k,l}}$. To which we find again a solution (the mysterious lucky heuristic) —in one second as expected— then repeat exactly 97 times (always the lucky heuristic) to reach a multiplication friendly lift over $R_{100}(4)$.

5 Applications to MPC

5.1 Proof of Main Theorem 2

Proposition 16. *For any fixed p and ℓ, consider any fix number n of players, and choose any fixed even integer r such that $p^r \geq 64$, and security parameter κ. Then there exists a slowly growing* infinite *sequence of integers N such that: for any set of N triples a_i, b_i, c_i in $\mathbb{Z}/p^\ell\mathbb{Z}$ (resp. in $\mathbb{F}_p$), which are shared between the players using any linear secret sharing scheme, then there exists a protocol that has the following properties*

- *The protocol consumes an additional number of triples, which is asymptotically $N(1 + 2p^{-r/2})$, that are opened (so cannot be used anymore);*
- *Either all triples considered are correct: $a_i b_i = c_i$ then it outputs* true*, or at least one is incorrect, then it outputs* false *except with probability $O(p^{-(r\kappa-1)/2}(1 + 4p^{-\kappa/2}))$*
- *The communication complexity is $nr(N+2\kappa^2)$ of elements of $\mathbb{Z}/p^\ell\mathbb{Z}$ (resp. of $\mathbb{F}_p$) sent, the computational complexity is $O(N)$ linear operations in $\mathbb{Z}/p^\ell\mathbb{Z}$ (resp. in $\mathbb{F}_p$) per player.*

For simplicity we prove it over finite fields. Then the same methods to lift it over rings as in Main Theorem 1 apply. Consider the finite field extension $\mathbb{F}_{p^r}$ and an optimal family of algebraic curves over $\mathbb{F}_{p^r}$ with genera slowly growing to infinity. The best existing asymptotic ratio of the number of rational points divided by the genus, is denoted $A(p^r)$ "the Ihara constant". When r is *even* then

Ihara showed existence of infinitely many curves with slowly growing genera such that it matches the upper-bound bound of Drinfeld-Vladuts: $A(p^r) = p^{r/2} - 1$. Recall that this bound is one order of magnitude lower than the Weil upper bound (which is relevant only for finite genera). Fix a curve $\mathcal{C}$ in this family, with genus $\mathfrak{g}$, such that it has at least $2(N + 2\mathfrak{g} - 1)$ points (which is possible for all N large enough since $p^r \geq 64$). Consider a fixed set of points $P_1, \ldots, P_N$ on this curve, and G a point of degree $N + 2\mathfrak{g} - 1$ (existence is guaranteed by [Sti09, Theorem 5.2.10 c)]). Then there exists an interpolation formula with coefficients linear in the a_i (resp. the b_i), that builds rational functions f (resp. g) in the Riemann Roch space $\mathcal{L}(G)$, such that they take the values a_i (resp. b_i) at the points $P_1, \ldots, P_N$. [The technique for this is as in Lagrange's interpolation formula: one considers for every point P_i a fixed public function χ_i that vanishes at all the P_j for $j \neq i$ but not at P_i. Existence of χ_i is guaranteed by a consequence of the Riemann-Roch formula: $\ell(G - \sum_{i \neq j} P_j) - \ell(G - \sum_j P_j) > 0$. Then, the function f is deduced as the linear combination $\sum_i (a_i / \chi_i(P_i)) f_i$]. The players can thus obtain a secret sharing of coefficients of f seen as a linear combination of the public χ_i's (same for g). Define $h = fg$ in $\mathcal{L}(2G)$. Consider the remaining points of the curve: $P_{N+1}, \ldots, P_{2(N+2\mathfrak{g}-1)}$. Players sacrifice $N + 2\mathfrak{g} - 1$ auxiliary triples (possibly incorrect), in order to compute with the Beaver passively-secure protocol (so possibly incorrectly) secret sharings of the products $\widetilde{c_i} = f(P_i)g(P_i)$ at all those remaining points. At this point, if all triples are correct and no cheating occured, then we should have $h(P_i) = c_i$ for all $i = 1 \ldots N$ and $h(P_i) = \widetilde{c_i}$ for all $i = N + 1 \ldots 2(N + 2\mathfrak{g} - 1)$ As above, players compute a secret sharing of the unique function $\widetilde{h}$ in $\mathcal{L}(2G)$ such that $\widetilde{h}(P_i) = c_i$ for all $i = 1 \ldots N$ and $\widetilde{h}(P_i) = \widetilde{c_i}$ (namely they locally compute a secret sharing of the coefficients of the linear decomposition of $\widetilde{h}$ along the public χ_i). Then they sample a random secret shared challenge value $\lambda \in \mathbb{F}_{p^{r\kappa}}$, compute locally secret sharings of the evaluations $f(\lambda), g(\lambda)$ and $\widetilde{h}(\lambda)$, compute a (possibly false) secret sharing of the product $f(\lambda)g(\lambda)$ by sacrificing $2\kappa^2$ triples (multiplication in $\mathbb{F}_{p^{r\kappa}}/\mathbb{F}_{p^r}$ being done with the schoolboy algorithm), then perform a equal-to-zero check on $f(\lambda)g(\lambda) - \widetilde{h}(\lambda)$. If it passes, then they return accept.

5.2 Existence of lifts of RMFE over rings, with constant rate

Let p be a prime and $r,\ k,\ m \geq 1$ be positive integers. $*$ denotes the componentwise product. We adapt over rings [Cas+18, Definition 1] (where $q = p^r$).

Definition 17. A pair (ϕ, ψ) is called an $(k, m)_{p^r}$ -Reverse Multiplication Friendly Embedding (RMFE) if $\phi : R_\ell(r)^k \to R_\ell(rm)$ and $\psi : R_\ell(rm) \to R_\ell(r)^k$ are two $R_\ell(r)$-linear maps satisfying

$$x * y = \psi(\phi(x)\phi(y)) \quad \text{for all } x, y \in R_\ell(r)^k \tag{19}$$

Theorem 18. *Consider the family of "Reverse multiplication friendly embeddings" (RMFE) of [Cas+18, Theorem 5] (where $q := p^r$), then there exists a family of RMFE of $(R_\ell(r))^k$ into $R_\ell(rm)$, with k slowly growing to infinity and the* same constant *asymptotic expansion rates m/k.*

Let us review the construction over fields of [Cas+18, Lem 6 & Cor 1] that provides [Cas+18, Theorem 5], and use the tools of §3.3 to show that it lifts. We consider a smooth curve over $\mathbb{F}_q$ of genus g, with k distinct rational points denoted $P_1, P_2, \ldots, P_k$. Let G be a divisor such that $\deg G \geq k + 2g + 1$ (and for simplicity, with support outside of $\{P_1, \ldots, P_k\}$). By the Riemann-Roch formula we thus have $\dim_{\mathbb{F}_q} L(G) - \dim_{\mathbb{F}q} L(G - \sum_i P_i) = k$. By §3.3, the Riemann Roch spaces in this equality lift to free modules of same rank. Consider the evaluation map $\pi : L(G) \longrightarrow \mathbb{F}_q^k : f \longrightarrow (f(P_i))_{i \in [k]}$, which has kernel $L(G - \sum_i P_i)$. Then π is surjective, since $\dim_{\mathbb{F}_q} \mathrm{Im}(\pi) = \dim_{\mathbb{F}_q} L(G) - \dim_{\mathbb{F}_q} L(G - \sum_i P_i) = k$. Surjectivity is preserved over rings (by the invertible determinant mod p trick).

Choose a subspace W of $L(G)$ of dimension k such that π induces an isomorphism between W and $\mathbb{F}_q^k$. Choose R a point of degree $m > 2\deg(G)$, which exists for m large enough by [Sti09, Theorem 5.2.10 c)]. For any $f \in L(G)$, we denote by $\boldsymbol{c_f}$ the evaluation vector $(f(P_i))$, and by $f(R)$ the evaluation. The previous isomorphism induces the $\mathbb{F}_q$-linear map $\phi : \pi(V) = \mathbb{F}_q^k \longrightarrow \mathbb{F}_{q^m} : \boldsymbol{c_f} \to f(R)$. Then ϕ is injective, since $\deg(R) > \deg(G)$. Thus the lift over rings is also injective, by Lemma 9.

Define the $\mathbb{F}_q$ -linear map $\tau : L(2G) \longrightarrow \mathbb{F}_{q^m} : f \to f(R)$. Then τ is injective, since $m = deg(R) > deg(2G)$, and likewise for the lift by Lemma 9. Bijectivity of $\mathrm{Im}(\tau)$ with $L(2G)$ induces the $\mathbb{F}_q$-linear map $\psi' : \mathrm{Im}(\tau) \subseteq \mathbb{F}_{q^m} \longrightarrow \mathbb{F}_q^k : f(R) \to (f(P_i))$. Then ψ' surjective (but not injective), by the same degree reason than π, and likewise for surjectivity of the lift. We extend ϕ' from $\mathrm{Im}(\tau)$ to all of $\mathbb{F}_{q^m}$ linearly, and denote the resulting map ψ.

Finally, RMFE follows from the fact that, for any $\boldsymbol{c_f}, \boldsymbol{c_g} \in \mathbb{F}_q^k$ we have:

$$\psi(\phi(\boldsymbol{c_f})\phi(\boldsymbol{c_g})) = \psi(f(R)g(R)) = \psi((f.g)(R)) = \boldsymbol{c_{fg}} = \boldsymbol{c_f} * \boldsymbol{c_g}$$

where $f, g \in W$ are uniquely determined from $\boldsymbol{c_f}$, $\boldsymbol{c_g}$ by the injectivities above. Note that $(fg)(R)$ belongs to $\mathrm{Im}(\tau)$ since $fg \in L(2G)$.

5.3 Proof of Main Theorem 3

We can now compile a protocol for a circuit over a large Galois ring $R_\ell(r)$, into a protocol for many evaluations in parallel of this circuit in $\mathbb{Z}/p^\ell\mathbb{Z}$ by casting over rings the protocols of [Cas+18]. Since we choose to restrict ourselves to the case of optimal adversary rate, we really need hyperinvertible matrices over Galois rings for any number of players (not the alternative with suboptimal adversary bound discussed in [Cas+18, §2.4]). Fortunately their construction is straightforward, see e.g. [Abs+19a]. We can thus cast the original protocol of Beerliova-Hirt over Galois rings, then compensate their bad asymptotic communication overhead by amortizing it over several instances in parallel, exactly as done in [Cas+18, Theorem 1 & 2]. Namely, the main tool are RMFE over rings with asymptotically *linear* rate, which is solved above in §5.2. Whereas the "tensoring-up" trick carries over rings without any technical difficulty.

5.4 An analogous efficient Hensel lift for RMFE

Again we consider for simplicity only the base field $\mathbb{F}_p$, instead of $\mathbb{F}_{p^r}$. Let us make the following useful rephrasing of the definition of a *reverse multiplication embedding* (RMFE) of $\mathbb{F}_p^k$ into $\mathbb{F}_{p^m}$ Consider the field extension $\mathbb{F}_{p^m}$, equipped with its internal multiplication law. Denoting the dual over $\mathbb{F}_p$ with $*$, this law is captured by what is denoted as the *multiplication tensor* $T \in \mathbb{F}_{p^m}^* \otimes \mathbb{F}_{p^m}^* \otimes \mathbb{F}_{p^m}$ Its components $T_{i=1..m}$ are $\mathbb{F}_p$-bilinear forms from $\left(\mathbb{F}_{p^m} \times \mathbb{F}_{p^m}\right)$ to $\mathbb{F}_p$. Now fix a linear map

$$\phi : \mathbb{F}_p^k \longrightarrow \mathbb{F}_{p^m}$$

The pull back of T :

$$\phi^* T = T(\phi(.), \phi(.))$$

decomposes in $\mathbb{F}_{p^m}$ in m components which are symmetric bilinear forms

$$\phi^* T_i = T_i(\phi(.), \phi(.)) \ , i = 1..m$$

belonging by definition to the symmetric tensor space of the linear forms $S^2((\mathbb{F}_p^k)^*)$.

Definition 19. Consider the (nonintegral) algebra $\mathbb{F}_p^k$, equipped with the multiplication law component-by-component. This law is captured by what is denoted as the "multiplication tensor", belonging to $\left(\mathbb{F}_p^k\right)^* \otimes \left(\mathbb{F}_p^k\right)^* \otimes \mathbb{F}_p^k$. We say that ϕ is a reverse multiplication embedding iff these m bilinear forms $\phi^* T_i$ generate the components $(x_1^* \otimes x_1^*, ...x_k^* \otimes x_k^*)$ of the multiplication tensor.

Lifting of an algorithm ϕ modulo p^2: Suppose we are given a reverse multiplication friendly embedding ϕ, over $\mathbb{F}_p$ ($r = 1$ to make notations simple): for each $j = 1 \dots k$, we have coefficients $\lambda_{i,j}$ such that:

$$x_j^* \otimes x_j^* = \sum_{i=1}^{m} \lambda_{i,j} . \phi^* T_i \tag{20}$$

(it is a tensorial equality: it takes place in the space of symmetric bilinear forms of length k, so expands on coordinates as a set of $k(k+1)/2$ equations). We want to lift ϕ and the coefficients $\lambda_{i,j}$ such that the equalities (20) hold modulo p^2. (So we have $mk + mk$ unknowns and m equations, each of them taking place in a symmetric tensor space of dimension $k(k+1)/2$). Consider arbitrary lifts ϕ' and $\lambda'_{i,j}$ of ϕ and $\lambda_{i,j}$ over $\mathbb{Z}/p^2\mathbb{Z}$, we thus obtain the (tensorial) equalities modulo p^2 for $j = 1..k$:

$$x_j^* \otimes x_j^* = \sum_{i=1}^{m} \lambda'_{i,j} \phi'^* T_i + p\Delta_j$$

and we would like to eliminate the error terms $p\Delta_j$ modulo p^2 by choosing better lifts of ϕ and of $\lambda_{i,j}$:

$$\phi' + p\psi \text{ and } \lambda'_{i,j} + p\mu_{i,j} \tag{21}$$

After replacing (21) in (20) then simplification, the equation becomes the following (tensorial) *linear* equation modulo p (so with coordinates in $\mathbb{F}_p$):

$$\sum_{i=1}^{m} 2\lambda'_{i,j} T_i\big(\phi'(.), \psi(.)\big) + \mu'_{i,j} T_i\big(\phi'(), \phi'_i(.)\big) = -\Delta_j$$

where the unknowns are ψ and $\mu'_{i,j}$.

How to repeat and compute higher lifts modulo p^ℓ then proceeds as in §4.

6 Ackowledgements

Matthieu Rambaud would like to thank Luc Illusie, Alberto Arabia, Stéphane Ballet, Mark Abspoel and Alain Couvreur.

References

[Abs+19a] Mark Abspoel, Ronald Cramer, Ivan Damgård, Daniel Escudero, and Chen Yuan. "Efficient Information-Theoretic Secure Multiparty Computation over $\mathbb{Z}/p^k\mathbb{Z}$ via Galois Rings". In: *TCC*. 2019.

[Abs+19b] Mark Abspoel, Anders Dalskov, Daniel Escudero, and Ariel Nof. *An Efficient Passive-to-Active Compiler for Honest-Majority MPC over Rings*. to appear in ACNS'21. 2019.

[Abs+20] M. Abspoel, R. Cramer, C. Yuan, I. Damgard, D. Escudero, M. Rambaud, and C. Xing. "Asymptotically Good Multiplicative LSSS over Galois Rings and Applications to MPC over $\mathbb{Z}_{2^k}$". In: *Asiacrypt*. 2020.

[Ara+18] Toshinori Araki, Assi Barak, Jun Furukawa, Marcel Keller, Yehuda Lindell, Kazuma Ohara, and Hikaru Tsuchida. "Generalizing the SPDZ Compiler For Other Protocols". In: *CCS*. 2018.

[BFO12] Eli Ben-Sasson, Serge Fehr, and Rafail Ostrovsky. "Near-Linear Unconditionally-Secure Multiparty Computation with a Dishonest Minority". In: *CRYPTO*. 2012.

[BH08] Z. Beerliová-Trubíniová and M. Hirt. "Perfectly-Secure MPC with Linear Communication Complexity". In: *TCC*. 2008.

[BMN18] Alexander R. Block, Hemanta K. Maji, and Hai H. Nguyen. "Secure Computation with Constant Communication Overhead Using Multiplication Embeddings". In: *Indocrypt*. 2018.

[Cas+09] I. Cascudo, H. Chen, Ronald Cramer, and C. Xing. "Asymptotically Good Ideal Linear Secret Sharing with Strong Multiplication over Any Fixed Finite Field". In: *CRYPTO*. Springer, 2009.

[Cas+18] Ignacio Cascudo, Ronald Cramer, Chaoping Xing, and Chen Yuan. "Amortized Complexity of Information-Theoretically Secure MPC Revisited". In: *CRYPTO*. Springer, 2018.

[Cas16] Ignacio Cascudo. "Secret Sharing Schemes with Algebraic Properties and Applications". In: *Pursuit of the Universal*. 2016.

[CC06] Hao Chen and Ronald Cramer. "Algebraic Geometric Secret Sharing Schemes and Secure Multi-Party Computations over Small Fields". In: *CRYPTO*. Springer, 2006.

[CCX11] Ignacio Cascudo, Ronald Cramer, and Chaoping Xing. "The Torsion-Limit for Algebraic Function Fields and Its Application to Arithmetic Secret Sharing". In: *CRYPTO*. 2011.

[CDM00] Ronald Cramer, Ivan Damgård, and Ueli Maurer. "General Secure Multi-party Computation from Any Linear Secret-sharing Scheme". In: *EUROCRYPT*. 2000.

[CDN15] Ronald Cramer, Ivan Bjerre Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. New York, NY, USA: Cambridge University Press, 2015.

[CG20] Ignacio Cascudo and Jaron Skovsted Gundersen. "A Secret-Sharing Based MPC Protocol for Boolean Circuits with Good Amortized Complexity". In: *TCC*. 2020.

[Cra+18] R. Cramer, I. Damgård, D. Escudero, P. Scholl, and C. Xing. "SPD$\mathbb{Z}_{2^k}$: Efficient MPC mod 2^k for Dishonest Majority". In: (2018).

[CS95] A. R. Calderbank and N. J. A. Sloane. "Modular and p-adic cyclic codes". In: *Designs, Codes and Cryptography* (1995).

[Dam+06] Ivan Damgård, Matthias Fitzi, Eike Kiltz, Jesper Buus Nielsen, and Tomas Toft. "Unconditionally Secure Constant-Rounds MPC for Equality, Comparison, Bits and Exponentiation". In: *TCC*. 2006.

[Dam+12] Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. "Multiparty Computation from Somewhat Homomorphic Encryption". In: *Advances in Cryptology – CRYPTO 2012*. 2012.

[Dam+19] I. Damgård, D. Escudero, T. Frederiksen, M. Keller, P. Scholl, and N. Volgushev. "New Primitives for Actively-Secure MPC over Rings". In: *IEEE S&P*. 2019.

[DLN19] Ivan Damgård, Kasper Green Larsen, and Jesper Buus Nielsen. "Communication Lower Bounds for Statistically Secure MPC, With or Without Preprocessing". In: *CRYPTO*. 2019.

[DLS20] Anders Dalskov, Eysa Lee, and Eduardo Soria-Vazquez. "Circuit Amortization Friendly Encodingsand Their Application to Statistically Secure Multiparty Computation". In: *ASIACRYPT*. 2020.

[DOS18] Ivan Damgård, Claudio Orlandi, and Mark Simkin. "Yet Another Compiler for Active Security or: Efficient MPC Over Arbitrary Rings". In: *CRYPTO*. 2018.

[FY92] Matthew Franklin and Moti Yung. "Communication Complexity of Secure Computation (Extended Abstract)". In: *STOC*. 1992.

[GS99] V. Guruswami and M. Sudan. "Improved decoding of Reed-Solomon and algebraic-geometry codes". In: *IEEE Trans. Inf. Theory* (1999).

[GSZ20] Vipul Goyal, Yifan Song, and Chenzhi Zhu. "Guaranteed Output Delivery Comes Free in Honest Majority MPC". In: *Advances in Cryptology – CRYPTO 2020*. 2020.

[Hes02] F. Hess. "Computing Riemann—Roch Spaces in Algebraic Function Fields and Related Topics". In: *J. Symb. Comput.* (2002).

[Ill05] Luc Illusie. "Grothendieck's existence theorem in formal geometry". In: *Fundamental Algebraic Geometry: Grothendieck's FGA Explained*. Ed. by AMS. 2005.

[Ish+07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. "Zero-knowledge from Secure Multiparty Computation". In: *STOC*. 2007.

[Khu04] Kamal Khuri-Makdisi. "Linear Algebra Algorithms for Divisors on an Algebraic Curve". In: *Math. Comput.* 73.245 (2004).

[NW17] Anand Kumar Narayanan and Matthew Weidner. "Subquadratic time encodable codes beating the Gilbert-Varshamov bound". In: *CoRR* (2017). URL: `http://arxiv.org/abs/1712.10052`.

[PC] İ. Keskinkurt Paksoy and M. Cenk. *TMVP-based Multiplication for Polynomial Quotient Rings*. eprint 2020/1302.

[PS20] Arpita Patra and Ajith Suresh. "BLAZE: Blazing Fast Privacy-Preserving Machine Learning". In: *NDSSS* (2020).

[PS21] Antigoni Polychroniadou and Yifan Song. "Constant-Overhead Unconditionally Secure Multiparty Computation Over Binary Fields". In: *Eurocrypt*. 2021.

[RB89] T. Rabin and M. Ben-Or. "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority". In: *STOC*. 1989.

[SAS17] M. Shi, A. Alahmadi, and P. Solé. *Codes and Rings*. Academic Press, 2017.

[SG20] P.-J. Spaenlehauer and A. le Gluher. "A fast randomized geometric algorithm for computing Riemann-Roch spaces". In: *Mathematics of Computation* (2020).

[SGA1] Alexander Grothendieck. *SGA 1*. LNM 224. Springer, 1964.

[Shu+01] K. W. Shum, I. Aleshnikov, P. V. Kumar, H. Stichtenoth, and V. Deolalikar. "A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert-Varshamov bound". In: *IEEE Trans. Inf. Theory* (2001).

[Sti09] H. Stichtenoth. *Algebraic Function Fields and Codes, 2nd edition*. Springer, 2009.

[SW99] M. A. Shokrollahi and H. Wasserman. "List decoding of algebraic-geometric codes". In: *IEEE Trans. Inf. Theory* (1999).

[Wal] Judy Walker. "Algebraic geometry codes over rings". PhD thesis. Univ Illinois Champain.

[Wal99] Judy L. Walker. "Algebraic geometric codes over rings". In: *Journal of Pure and Applied Algebra* 144.1 (1999), pp. 91–110.