Integer Secret Sharing Using CRT

Tejas Balasaheb Pujari

CrS2222

Supervisors:

Prof. Ivan Damgård
Aarhus University, Denmark

Prof. Mridul Nandi
Indian Statistical Institute, Kolkata

Master of Technology in Cryptology & Security

at the

Indian Statistical Institute, Kolkata

Abstract

In this work, we revisit Yao's [Yao82] celebrated 1982 question concerning the collaborative computation of integer functions by a set of n parties, each initially possessing only their respective inputs. The challenge is to compute an integer function without revealing individual inputs. Previous solutions typically assume f is represented by an arithmetic circuit over a finite field, limiting applicability to integer-based functions as originally proposed by Yao. Adapting $\mathbb{F}$ to simulate integer computations introduces practical issues: the need for known input bounds and the limitations of finite field arithmetic compared to integers. In an ongoing MPC computation, the input can be provided in any round and the input size might depend on unpredictable factors. Hence, the size of the field might be impossible to predict or must be chosen unreasonably big to simulate an integer computation of a function f.

In this thesis, we introduce a novel non-linear integer secret-sharing scheme tailored specifically for integer secrets. Unlike traditional approaches, which often rely on finite fields, our scheme operates directly over integers, leveraging the Chinese Remainder Theorem to design a ramp secret-sharing scheme. This approach combines the efficiency of modular arithmetic with the inherent security properties of secret sharing. The ramp scheme enables efficient reconstruction of secrets while providing statistical privacy guarantees, ensuring robust protection of sensitive information.

Acknowledgements

First and foremost, I extend my deepest gratitude to Ivan Damgård for his invaluable supervision throughout this journey. His unwavering availability for discussions on cryptography, generous sharing of knowledge and insights, and constant inspiration for new avenues of exploration have been instrumental in the completion of this work. Without his guidance and support, this thesis would not have been possible.

I am equally thankful to my co-supervisor, Mridul Nandi, for his steadfast support and confidence in my research. His guidance and encouragement have been invaluable in shaping this thesis.

I am grateful to the Aarhus Crypto group and researchers from the Complexity group for their stimulating discussions and valuable insights into my research.

Lastly, I owe a heartfelt thank you to my family for their unwavering support and encouragement throughout this journey.

Prof. Mridul Nandi

Tejas Pujari
June 30, 2024

Contents

1 Introduction

Cryptography empowers mutually distrusting parties to collaborate effectively, provided a subset of participants remain honest. A cornerstone of this capability is Secure Multi-Party Computation (MPC) [Yao86, GMW87], allowing parties to jointly compute public functions over private inputs without revealing more than the function output, contingent upon a subset of participants behaving honestly. Specific applications like threshold signatures [Des88, DF90] and encryption schemes distribute secret keys among parties, enabling actions such as signing messages or decrypting ciphertexts if a threshold number of parties participate honestly.

This paradigm has proven highly successful, with threshold cryptosystems becoming integral to blockchain ecosystems [SC17], prompting efforts toward standardization [Ost91]. Recent advancements in MPC protocols have significantly enhanced their efficiency, driving broader adoption.

Initially proposed by Yao [Yao82], the goal of MPC is to enable a set of n parties $P = \{P_1, \ldots, P_n\}$ to compute an integer-valued function $f(x_1, \ldots, x_n)$, where each x_i is an integer variable of bounded range. Each party P_i initially knows only its input x_i. The challenge is to compute f through communication without divulging individual inputs, ensuring correctness despite the potential corruption of some parties by malicious adversaries. Early research has focused on optimizing communication and round complexities [Yao82, GMW87, FY92, Genn98, Hirt97], and addressing more powerful or generalized adversaries [Yao86, GMW87, Ost91, Can96, CDM00, Fitzi97].

Integer secret-sharing schemes play a crucial role in solving distributed exponentiation problems inherent in cryptographic protocols such as RSA encryption. Despite their significance, the predominant approach in the literature remains limited to linear integer Secret-Sharing Schemes (LISS). LISS performs all arithmetic operations directly over integers without any involvement of fields. This method, while comprehensive in handling integer arithmetic, is known for its inefficiency in terms of computational complexity.

This leads to a natural question to ask:

Can we realize integer secret sharing more efficiently?

Summary Our work provides an affirmative answer. We propose a novel and efficient non-linear threshold ramp secret-sharing scheme for integer secrets.

1.1 Our Contribution: Non-Linear Threshold Ramp Secret Sharing

In this thesis, we introduce a ramp scheme specifically designed for sharing integer secrets. The secret s is selected from known intervals, for example, $[2^{-(\sigma-1)}, 2^{\sigma-1}] \subset \mathbb{Z}$. Let $P = \{P_1, P_2, \ldots, P_n\}$ denote the set of n parties among whom the secret s is shared. The shares are computed as a large integer S, which masks the secret s modulo primes assigned to each party. The scheme is characterized by a privacy threshold t and a reconstruction threshold T.

Scheme Description Given a qualified set $Q \subset P$ of parties and the associated shares $\{s_i\}_{i \in Q}$, the secret S is reconstructed as an integer linear combination of the shares:

$$S = \sum_{j \in Q} \lambda_j \cdot s_j \mod \left(\prod_{j \in Q} p_j \right)$$

$$s = S \mod 2^{\sigma}$$

Scheme Properties The proposed non-linear threshold ramp secret sharing scheme has the following properties:

1. **Share Size**: The share size for each party P_i is $O(\log(p_i))$.

2. **Perfect Correctness**: The scheme guarantees perfect correctness.

3. **Statistical Privacy**: The scheme achieves 2^{λ} statistical privacy.

This scheme effectively balances efficiency and security, making it suitable for practical applications where integer secrets need to be securely shared and reconstructed.

1.2 Efficient MPC Protocol Based on the BGW Framework

Following this work, we develop an efficient MPC protocol based on the BGW framework using our scheme for Secure Multiparty Computation. We aim for information-theoretic security in the honest majority setting. Using our new scheme, we construct an MPC protocol following the BGW framework [BGW88]. Our result is summarized as follows.

Theorem 1. *Efficient MPC*
Let C be an arithmetic circuit with depth d over $\mathbb{Z}$. There exists an MPC protocol for n parties for computing C, satisfying the following:

1. ***Round Complexity**: The round complexity is d.*
2. ***Preprocessing Phase**: The communication required in this phase is $O(\sum_i \log(p_i))$.*
3. ***Protocol Run Phase**: During this phase, the communication required is $O(\max\{\sigma, \log(p_i)\})$, where σ is public parameter relates to secret space.*
4. ***Reconstruction Phase**: The communication required in this phase is $O(\sum_i \log(p_i))$.*

The protocol is $2^{-\lambda}$-secure.

1.3 Efficiency Comparison with Existing Schemes

We conclude this work by providing an efficiency comparison of our scheme with existing schemes. This comparison highlights the advantages of our approach in terms of communication complexity and security guarantees.

1.4 Previous Work

CRT-Based Secret Sharing Secret sharing schemes based on the Chinese Remainder Theorem (CRT) were initially proposed by Mignotte [Mig83] and later by Asmuth and Bloom [AB83]. Further advancements were made by Iftene and Boureanu [IB05], Mignotte's approach to incorporate weighted settings. However, their method essentially employs a straightforward virtualization technique for CRT-based secret sharing.

Ramp Secret Sharing Ramp secret sharing was introduced by Blakley and Meadows [BM84] as a novel technique in the field of secret-sharing schemes. This approach generalizes the traditional secret-sharing model, offering a way to balance between share size and security requirements. Over time, Ramp secret sharing has been pivotal in the development of more efficient multiparty computation (MPC) protocols and other cryptographic applications. For instance, the concept of packing, introduced by Fehr and Yang [FY92], utilizes Ramp secret sharing to reduce the size of shares, which is crucial for the efficiency of MPC protocols [FY92, DIK+08, DIK10]. Additionally, Ramp secret sharing has found applications in broadcast encryption schemes, such as those explored by Sewell and Walker [SW99], which leverage the properties of Ramp schemes to achieve secure and efficient broadcasting solutions.

Non-Linear Secret Sharing In contrast to linear secret-sharing schemes, non-linear secret-sharing schemes are not as comprehensively understood. Most of the studied non-linear schemes are tailored for specific access structures [BI01] or generalized access structures [LV18, ABF+19, ABNP20].

Applications of Integer Secret Sharing Integer secret sharing finds natural applications in several cryptographic protocols. Notably, it can be employed in *distributed exponentiation* [DT06] and is instrumental in constructing *distributed threshold encryption schemes*. Additionally, integer secret sharing can be utilized for *secure multiparty computation*, *verifiable secret sharing* [DT06, EPKP+20], and other cryptographic primitives where distributed and fault-tolerant operations on secret data are required. These applications highlight the versatility and importance of integer secret sharing in modern cryptographic systems.

2 Preliminaries

2.1 Secret Sharing

Secret sharing is a cryptographic method where a dealer distributes shares of a secret among several shareholders such that only certain subsets of them, called *qualified sets*, can reconstruct the secret. Non-qualified subsets have no information about the secret. Collecting all qualified sets is known as the *access structure*. Secret sharing was first introduced to securely store critical information, ensuring privacy and protection against data loss.

2.1.1 Threshold Secret Sharing

Threshold secret sharing is a specific type of secret sharing where the access structure is defined as all subsets of parties whose size exceeds a certain threshold t. More formally, a threshold (t,n) secret sharing scheme over some secret space $\mathbb{Z}$ consists of two efficient algorithms: **Share** and **Reconstruct**.

- **Share**: A probabilistic algorithm $\text{Share}(N,t,s) \to (s_1,\ldots,s_N)$, where N is the number of parties, t is the threshold $(0 < t < N)$, and s is the secret to be shared.
- **Reconstruct**: A deterministic algorithm $\text{Reconstruct}(J,\{s_j\}_{j\in J}) \to s$, where J is a subset of parties with size greater than t, enabling the reconstruction of the secret s.

The access structure realized by this secret-sharing scheme is called the threshold-t structure.

2.1.2 Security Guarantees

Secret-sharing schemes provide two key guarantees:

- **Correctness**: Ensures that the secret can be correctly reconstructed by any qualified set of parties.
- **Privacy**: Guarantees that a set of unauthorized parties (those not in access structure) gain no information about the secret.

For a threshold secret sharing scheme with threshold parameter t:

Correctness: For any subset J of parties with size greater than t, for any secret $s \in \mathbb{Z}$, and for any sharing output $(s_1,\ldots,s_N)$ generated by $\text{Share}(N,t,s)$, it holds that

$$\text{Reconstruct}(J,\{s_j\}_{j\in J}) = s.$$

Privacy: Any subset of t or fewer parties must not gain any information about the secret s. Formally, the views of an adversary controlling any t parties regarding two different secrets s and s' should be indistinguishable:

$$\text{view}_s^A \approx \text{view}_{s'}^A,$$

where

$$\text{view}_s^A = \left\{ \begin{array}{c} \{s_{i_1}, s_{i_2}, \ldots, s_{i_t}\} \\ | \\ s_{i_j} \text{ is the share of Party } P_{i_j}, \\ \{P_{i_j}\}_{j=1}^t \text{ is the set of parties controlled by Adversary } A \end{array} \right\}.$$

2.1.3 Shamir Secret Sharing

Shamir's Secret Sharing is a cryptographic method introduced by Adi Shamir in 1979,[Sha79] which allows a secret to be divided into parts, giving each participant a unique part. More than a threshold number of parts is required to reconstruct the secret. This method is based on polynomial interpolation and offers security and flexibility, making it one of the most celebrated threshold secret-sharing schemes.

2.1.4 (t,n) Shamir Secret Sharing Protocol

Initialization

- n: Number of parties.
- F: Finite field such that $|F| > n$.
- $s \in F$: Secret to be shared.
- $\alpha_1, \alpha_2, \ldots, \alpha_n$: Distinct non-zero public evaluation points.
- D: Dealer.

Sharing

- **Secret and Random Points:** The dealer D selects the secret s and chooses t random coefficients $a_1, a_2, \ldots, a_t$ from the field F.
- **Polynomial Definition:** Define the polynomial:

$$f(x) = s + a_1 x + a_2 x^2 + \ldots + a_t x^t$$

- **Shares Distribution:** Each party P_i receives a share $f(\alpha_i)$ for $i = 1, 2, \ldots, n$.

Reconstruction To reconstruct the secret s:

- **Lagrange Interpolation:** Use the shares from any $t+1$ parties. The polynomial $f(x)$ can be reconstructed using Lagrange interpolation:

$$f(x) = \sum_{j=1}^{t+1} \lambda_j f(\alpha_j)$$

where

$$\lambda_j = \prod_{i \neq j} \frac{(x - \alpha_i)}{(\alpha_j - \alpha_i)}$$

- **Secret Extraction:** The secret is the constant term of the polynomial:

$$s = f(0)$$

2.1.5 Security of Shamir's Scheme

Shamir's Secret Sharing scheme is information-theoretically secure, which means that an adversary with fewer than t shares learns nothing about the secret. The security relies on the following properties:

- **Polynomial Degree:** The polynomial $f(x)$ is of degree t. Knowing fewer than $t+1$ points provides no information about the polynomial's coefficients due to the properties of polynomial interpolation over a finite field.
- **Random Coefficients:** The coefficients $a_1, a_2, \ldots, a_{t-1}$ are chosen randomly from the field F, ensuring that any set of $t+1$ shares can reconstruct the polynomial, Still any set of fewer than $t+1$ shares provides no information about s.

Overall, Shamir's Secret Sharing scheme ensures that only a coalition of at least $t+1$ parties can reconstruct the secret, making it a robust and secure method for secret distribution and recovery.

2.2 Ramp Threshold Secret Sharing

While threshold schemes offer a robust method for secret sharing, they sometimes lack the efficiency required in certain applications. This has led to the development of the ramp model, a natural extension of the threshold scheme. The ramp model introduces two thresholds: a privacy threshold and a reconstruction threshold, enhancing flexibility and efficiency.

In a ramp threshold secret sharing scheme:

- Define t as the privacy threshold and T as the reconstruction threshold, satisfying $t < T$.
- Any subset of parties J with $|J| \leq t$ should not gain any information about the secret s.
- Any subset of parties J with $|J| \geq T$ can fully reconstruct the secret s with perfect accuracy.
- Subsets with cardinality between $t+1$ and $T-1$ may have partial or complete information about the secret s.

The ramp scheme provides a balance between security and efficiency:

- **Improved Efficiency**: By allowing subsets of parties to have partial information, the ramp scheme reduces the overall amount of data each party needs to store and process, improving the efficiency of the scheme.
- **Flexibility**: The dual-threshold approach allows for fine-tuning of security and reconstruction parameters, making the scheme adaptable to different levels of security requirements and resource constraints.
- **Enhanced Scalability**: The ramp scheme is particularly useful in scenarios where the number of parties is large, and a strict threshold scheme would be too rigid or inefficient.

The introduction of two thresholds in the ramp model thus offers a practical and versatile alternative to traditional threshold secret sharing, addressing both the need for strong privacy guarantees and the demand for operational efficiency.

2.3 Random Variables

Let Ω denote the collection of all outcomes for a given experiment.

Definition 1. *A collection $\mathscr{A}$ of subsets of Ω is an **algebra** if:*

1. *$A, B \in \mathscr{A}$ implies $A \cup B \in \mathscr{A}$.*
2. *$A \in \mathscr{A}$ implies $A^{\complement} \in \mathscr{A}$.*
3. *$\Omega \in \mathscr{A}$.*

Definition 2. *A collection $\mathscr{F}$ of subsets of Ω is a **σ-algebra** if:*

1. *$\mathscr{F}$ is an algebra.*
2. *If $\{A_j\}$ is an infinite sequence in $\mathscr{F}$, then $\bigcup A_j \in \mathscr{F}$.*

Definition 3. *A **probability space** is a triple $(\Omega, \mathscr{F}, \Pr)$ where $\mathscr{F}$ is a nonempty σ-algebra of subsets of Ω and $\Pr$ is a mapping from $\mathscr{F}$ to $\mathbb{R}$ satisfying:*

1. *$\Pr(\Omega) = 1$.*
2. *$0 \leq \Pr(A) \leq 1$ for all $A \in \mathscr{F}$.*

3. *If $\{A_j\}$ is a finite or countably infinite sequence of disjoint sets in $\mathscr{F}$, then*

$$\Pr\left(\bigcup A_j\right) = \sum \Pr(A_j).$$

Definition 4. *Let $(\Omega, \mathscr{F}, \Pr)$ be a probability space. A mapping $X : \Omega \to \mathbb{R}$ is called a **random variable** if for every real number x, the set $(X \leq x) = \{\omega \mid X(\omega) \leq x\} \in \mathscr{F}$. This means that the event "the random variable X takes a value less than or equal to x" is measurable.*

Example 1. *Let $\Omega = \left[-2^l, 2^l\right] \subset \mathbb{Z}$ and $X(\omega) = \omega$ for $\omega \in \Omega$. Then $(X \leq x) = \{\omega \mid X(\omega) \leq x\} = \{\omega \mid \omega \leq x\} \subset \mathbb{Z}$. That $(X \leq x) \in \mathscr{F}$ for all $x \in \mathbb{R}$ means that $\Pr(X \leq x)$ is defined for all $x \in \mathbb{R}$ and defines a function on $\mathbb{R}$.*

If the domain of a random variable is a countable set $\{x_1, x_2, \ldots\}$, then it follows that $\{x_i\} \in \mathscr{F}$, hence, $\Pr(X = x_i)$ is well defined for all $i = 1, 2, \ldots$.

We will only consider countable finite random variables, similar to the one given in the example above.

Definition 5. *Let X and Y be countable random variables taking values in set V. The **statistical distance** is defined as:*

$$[X;Y] = \frac{1}{2} \sum_{v \in V} |\Pr(X = v) - \Pr(Y = v)|.$$

Example 2. *Let X be a random variable with the domain $\{1, 2, \ldots, n\}$ and define the random variable $Y = X + m$ for a positive m, i.e., Y has domain $\{1+m, 2+m, \ldots, n+m\}$ and $\Pr(X = i) = \Pr(Y = i+m)$ for $i = 1, 2, \ldots, n$.*

First, observe that if $n < m+1$, then the statistical distance $[X;Y] = 1$. Assume that $m < n$ and that X has a uniformly random distribution, then the statistical distance is given by:

$$[X;Y] = \frac{m}{n}.$$

Theorem 2 (Union Bound)**.** *If $\{A_i\}$ is any sequence of events, then*

$$\Pr\left[\bigcup_i A_i\right] \leq \sum_i \Pr[A_i].$$

Definition: Data Processing Inequality (DPI): Let X, Y, and Z be random variables such that $X \to Y \to Z$ forms a Markov chain. For any function f such that $X \to Y \to Z$ and $f(Y) = Z$, the DPI states:

$$I(X;Z) \leq I(X;Y),$$

where $I(X;Y)$ denotes the mutual information between X and Y, and $I(X;Z)$ denotes the mutual information between X and Z.

Components:

- **Random Variables**: X, Y, and Z are random variables representing data or information at different stages of processing.
- **Markov Chain**: The sequence $X \to Y \to Z$ indicates a Markov chain where Y acts as an intermediary between X and Z. This implies that Y captures all relevant information from X necessary for predicting Z.
- **Mutual Information**: $I(X;Y)$ measures how much information X and Y share. Similarly, $I(X;Z)$ measures the information shared between X and Z.
- **Inequality Statement**: The DPI asserts that the mutual information $I(X;Z)$, after X is processed through Y to produce Z, cannot exceed the mutual information $I(X;Y)$ between X and Y. In other words, Y acts as a bottleneck, limiting the amount of information that Z can retain from X.

2.4 Chinese Remainder Theorem

The Chinese Remainder Theorem is one of the oldest known theorems in number theory. The earliest known statement of the theorem, as a problem with specific numbers, is attributed to the Chinese mathematician Sun Zi in the 3rd century CE:

> There are certain things whose number is unknown. If we count them by threes, we have two left over; by fives, we have three left over; and by sevens, two are left over. How many things are there?

- The first known solution to this problem was provided by the mathematician Aryabhata in 6 CE.

Statement: Let $n_1, n_2, \ldots, n_r$ be pairwise coprime integers. Then the system of linear congruences

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} && (1)\\ x &\equiv a_2 \pmod{n_2} && (2)\\ x &\equiv a_3 \pmod{n_3} && (3)\\ &\vdots && (4)\\ x &\equiv a_r \pmod{n_r} && (5) \end{aligned}$$

admits a solution which is unique modulo $n_1 n_2 \cdots n_r$.

The general solution to the above system is given by:

$$x = a_1 c_1 + a_2 c_2 + \cdots + a_r c_r \pmod{N}, \tag{6}$$

where $N = n_1 n_2 \cdots n_r$ and c_i satisfies the following conditions:

- $c_i \equiv 1 \pmod{n_i}$,
- $c_i \equiv 0 \pmod{n_j}$ for all $j \neq i$.

Example: To solve the problem posed by Sun Zi, where we have the system of congruences:

$$\begin{aligned} x &\equiv 2 \pmod{3} && (7)\\ x &\equiv 3 \pmod{5} && (8)\\ x &\equiv 2 \pmod{7}, && (9) \end{aligned}$$

we use the Chinese Remainder Theorem to find x.

Solution: To find x, we need to solve these congruences. Since $3, 5$, and 7 are pairwise coprime, there is a unique solution modulo 105 (since $3 \times 5 \times 7 = 105$).

One approach is to compute:

- Compute $N = 3 \times 5 \times 7 = 105$.
- Find N_i for each i, where $N_i = N/n_i$.
- Find c_i such that $c_i \equiv 1 \pmod{n_i}$ and $c_i \equiv 0 \pmod{n_j}$ for all $j \neq i$.
- Combine results to find the solution x.

2.5 Adversarial structure and security of the protocol

In this section, we review the definition of perfect security and statistical security in the presence of **semi-honest** We refer the reader to [Can00] for more details and discussion. In the definitions below, we consider the **stand-alone** setting with a synchronous network and perfectly private channels between all parties. For simplicity, we assume that the parties have access to a broadcast channel, which can be implemented using an appropriate Byzantine Generals protocol [LSP82].

Since we consider synchronous channels and the computation proceeds in clearly defined rounds, if a message is not received in a given round, then this fact is immediately known to the party intended to receive the message. Thus, terms such as "if a message is not received" or "if the adversary does not send a message" are well-defined.

Notations $C : \mathscr{X}_1 \times \mathscr{X}_2 \times \ldots \times \mathscr{X}_n \longrightarrow \mathscr{Y}$ denotes the arithmetic circuit over some space. We denote $I = \{i_1, i_2, \ldots, i_t\} \subset [n]$, where $[n] = \{1, 2, \ldots, n\}$ are indices corresponding to corrupted parties. Let $\vec{x} = \{x_1, x_2, \ldots, x_n\}$ be the input vector, where x_i is the input of party i to the arithmetic circuit C. $\vec{x}_I$ and $C(\vec{x})_I$ denote the restriction of the input vector and output vector to set I, respectively. In this case, they are the same for all because we are restricting the functionality to a single output, but this generalizes to the multi-output setting. **S** denotes the simulator (adversary of the ideal world).

2.5.1 Perfect Security Definitions:

We are now ready to define security in the presence of semi-honest adversaries. Loosely speaking, the definition states that a protocol is t-semi-honest secure if the view of up to t corrupted parties in a real protocol execution can be generated by a simulator given only the corrupted parties' inputs and outputs.

The view of the i-th party P_i during an execution of a protocol π on inputs $\vec{x}$, denoted $\text{VIEW}_i^{\pi}(\vec{x})$, is defined as $(x_i, r_i; m_{i_1}, \ldots, m_{i_k})$ where x_i is P_i's private input, r_i is its internal coin tosses, and m_{i_j} is the j-th message received by P_i in the protocol execution. For every $I = \{i_1, \ldots, i_\ell\} \subseteq [n]$, we denote $\text{VIEW}_I^{\pi}(\vec{x}) = \left(\text{VIEW}_{i_1}^{\pi}(\vec{x}), \ldots, \text{VIEW}_{i_\ell}^{\pi}(\vec{x})\right)$. The output of all parties from an execution of π on inputs $\vec{x}$ is denoted $\text{OUTPUT}^{\pi}(\vec{x})$; observe that the output of each party can be computed from its own (private) view of the execution.

We first present the definition for deterministic functionalities, since this is simpler than the general case of probabilistic functionalities.

2.5.2 Perfect Security of n-Party Protocols - Deterministic Functionalities

Let C be a deterministic arithmetic circuit and let π be a protocol. We say that π is semi-honest perfectly secure for C if for every $\vec{x} \in \mathscr{X}_1 \times \mathscr{X}_2 \times \ldots \times \mathscr{X}_n$ where $|\vec{x}_1| = \ldots = |\vec{x}_n|$,

$$\text{OUTPUT}^{\pi}(\vec{x}) = C(\vec{x})$$

and there exists a probabilistic polynomial-time algorithm $\mathscr{S}$ such that for every $I \subseteq [n]$ with $|I| \leq t$,

$$\{\mathscr{S}(I, \vec{x}_I, C(\vec{x})_I)\} \equiv \{\text{VIEW}_I^{\pi}(\vec{x})\}.$$

The above definition separately considers the issue of output correctness and privacy, where the latter captures privacy since the ability to generate the corrupted parties' view given only the input and output means that nothing more than the input and output is learned from the protocol execution. However, in the case of probabilistic functionalities, it is necessary to intertwine the requirements of privacy and correctness and consider the joint distribution of the output of $\mathscr{S}$ and of the parties. Thus, in the general case of probabilistic functionalities, the following definition is used.

2.5.3 Perfect Security of n-Party Protocols - general case

Let $C : \mathscr{X}_1 \times \mathscr{X}_2 \times \ldots \times \mathscr{X}_n \longrightarrow \mathscr{Y}_1 \times \mathscr{Y}_2 \times \ldots \times \mathscr{Y}_n$ be a probabilistic n-ary arithmetic circuit and let π be a protocol. We say that π is semi-honest perfectly secure for C if there exists a probabilistic polynomial-time algorithm $\mathscr{S}$ such that for every $I \subseteq [n]$ with $|I| \leq t$,

$$\{(\mathscr{S}(I, \vec{x}_I, C(\vec{x})_I), C(\vec{x}))\} \equiv \{(\text{VIEW}_I^{\pi}(\vec{x}), \text{OUTPUT}^{\pi}(\vec{x}))\}.$$

2.5.4 Statistical Security

We now define statistical security for protocols against adversaries. Statistical security ensures that the output of the protocol execution remains indistinguishable from the ideal distribution, even when the adversary has access to auxiliary information.

Statistical security of n-party protocols Let $C : \mathscr{X}_1 \times \mathscr{X}_2 \times \ldots \times \mathscr{X}_n \longrightarrow \mathscr{Y}_1 \times \mathscr{Y}_2 \times \ldots \times \mathscr{Y}_n$ be an n-ary probabilistic functionality and let π be a protocol. We say that π is statistically semi-honest-secure for C if there exists a negligible function $\text{negl}(\cdot)$ such that for every $I \subseteq [n]$ with $|I| \leq t$,

$$\{(\mathscr{S}(I, \vec{x}_I, C(\vec{x})_I), C(\vec{x}))\} \approx_{\text{stat}} \{(\text{VIEW}_I^{\pi}(\vec{x}), \text{OUTPUT}^{\pi}(\vec{x}))\}.$$

where $\mathscr{S}$ is a probabilistic polynomial-time algorithm (simulator), $\vec{x} \in \mathscr{X}_1 \times \mathscr{X}_2 \times \ldots \times \mathscr{X}_n$ with $|\vec{x}_1| = \ldots = |\vec{x}_n|$, More formally the definition says that, A protocol π statistically securely realizes circuit C if for every adversary $\mathscr{A}$ corrupting a subset I of parties, there exists a probabilistic polynomial-time simulator S such that for any probabilistic polynomial-time distinguisher D:

$$|\Pr[D(\mathscr{S}(I, \vec{x}_I, C(\vec{x})_I)) = 1] - \Pr[D(\text{VIEW}_I^{\pi}(\vec{x})) = 1]| \leq \text{negl}(\lambda)$$

and λ denotes the statistical security parameter.
Our protocol based on the BGW framework will prove the protocol is statistically secure.

3 Integer Secret Sharing

In this chapter, we explore early constructions in integer secret sharing. We begin with the Linear Integer Secret Sharing (LISS) scheme. Cramer and Fehr introduced the concept of Integer Span Programs (ISPs), and we will examine the construction of LISS based on ISP by Damgard and Thorbek [DT06].

It has been established that achieving **perfect secret sharing** and private computation over countably infinite domains, such as integers, is fundamentally impossible [Chor93, Chor95]. However, this limitation does not extend to scenarios where secrets are constrained to publicly known intervals. In such cases, cryptographic schemes can still provide statistical privacy, which aims to limit the amount of information an adversary can gain about the secret, rather than guaranteeing absolute secrecy. This distinction allows for practical implementations where the focus shifts from unattainable perfect privacy to achievable statistical security measures.

The concept of Integer Span Programs (ISPs), introduced by Cramer and Fehr [CF00], plays a pivotal role in constructing cryptographic schemes like BBSS (Black box secret sharing)[DF94] schemes. ISPs are characterized by matrices with integer entries, which dictate how secrets and randomness are combined to generate shares. Importantly, ISPs facilitate BBSS schemes and serve as a foundation for developing linear integer secret sharing (LISS) schemes. This connection underscores that an ISP designed for threshold-t access structures can effectively translate into a LISS scheme, expanding the applicability of cryptographic tools beyond traditional perfect secrecy paradigms. Moreover, adaptations of monotone formulas by Benaloh and Leichter [BL88] further demonstrate the versatility of cryptographic constructions, ensuring that LISS schemes can be tailored to diverse access structures, albeit with considerations for computational efficiency.

Before diving into the scheme, we first define integer span programs.

3.1 Integer Span Program (ISP)

Definition 6. *$\mathscr{M} = (M, \psi, \varepsilon)$ is termed an Integer Span Program (ISP) if:*

- *$M \in \mathbb{Z}^{d \times e}$, where d denotes the number of rows of M.*
- *The rows of M are labeled by a surjective function $\psi : \{1, \ldots, d\} \to \{1, \ldots, n\}$.*
- *$\varepsilon = (1, 0, \ldots, 0)^T \in \mathbb{Z}^e$ is the target vector.*

The size of $\mathscr{M}$ is defined as d.

3.1.1 ISP for Access Structure Γ

Definition 7. *Let Γ be a monotone access structure and $\mathscr{M} = (M, \psi, \varepsilon)$ an ISP. $\mathscr{M}$ is an ISP for Γ if:*

- *For all $A \in \Gamma$, there exists a reconstruction vector $\lambda \in \mathbb{Z}^d$ such that $M_A^T \cdot \lambda = \varepsilon$.*
- *For all $A \notin \Gamma$, there exists a sweeping vector $\kappa \in \mathbb{Z}^e$ such that $M_A \cdot \kappa = \mathbf{0}$ and $\langle \kappa, \varepsilon \rangle = 1$.*

In other words, qualified sets A must include the target vector ε in their span, whereas forbidden sets A must have a sweeping vector κ orthogonal to all rows of M_A, but with $\langle \kappa, \varepsilon \rangle = 1$. This property implies that $\mathscr{M}$ computes the access structure Γ.

Define $\kappa_{\max} = \max\{|a| : \text{a is an entry in some sweeping vector}\}$. Let $l_0 = l + \lceil \log_2(\kappa_{\max}(e-1)) \rceil + 1$. The size of $\mathscr{M}$ is defined to be d.

3.2 Linear Integer Secret Sharing

To share a secret $s \in [-2^l, 2^l]$, a distribution vector ρ is used, which is uniformly random in $[-2^{l_0+k}, 2^{l_0+k}]^e$ with $\langle \rho, \varepsilon \rangle = s$, where k is the statistical security parameter. The share vector is computed as:

$$s = (s_1, \ldots, s_d)^T = M\rho,$$

where each share component s_i corresponds to party $P_{\psi(i)}$ for $1 \leq i \leq d$. The share of party P_j includes the subset of share components $s_{\{P_j\}}$.

The first requirement in the Definition ensures the correctness of the scheme, as a qualified set A can reconstruct the secret s by taking a linear combination of their shares. Specifically, there exists $\lambda_A \in \mathbb{Z}^{d_A}$ such that $M_A^T \lambda_A = \varepsilon$, leading to:

$$s_A^T \lambda_A = (M_A \cdot \rho)^T \lambda_A = \rho^T \left(M_A^T \lambda_A\right) = \rho^T \varepsilon = s.$$

Theorem 3. *If $s \in [-2^l, 2^l]$ is the secret to be shared and ρ is chosen uniformly at random from $[-2^{l_0+k}, 2^{l_0+k}]^e$ with the restriction that $\langle \rho, \varepsilon \rangle = s$, then the LISS scheme derived from $\mathscr{M}$ is private.*

We will not prove the statements above. The reader can find the more detailed work in [DT06] Instead, we move towards the efficiency of the LISS scheme derived above.

3.3 Comparison of LISS Schemes

We will compare two LISS schemes based on two constructions of ISP given by Cramer-Fehr and Benaloh-Leichter concerning the threshold t-scheme.

3.3.1 Share Sizes

Consider the share sizes in each construction of a threshold-t scheme $T_{t,n}$ with a secret of bit length at most l and statistical security parameter k:

Construction	Share size
Cramer-Fehr	$\mathscr{O}((l+k+n)\log n)$
Benaloh-Leichter	$\mathscr{O}((l+k+\log\log n)n^{\sqrt{2}})$

In a typical scenario, the parameters l and k dominate. Ignoring hidden constants, the slight advantage goes to the Cramer-Fehr construction.

3.3.2 Local Computation Time

Considering the local computation time to generate a set of shares, the advantage shifts to the Benaloh-Leichter construction:

Construction	Local computation time
Cramer-Fehr	$\mathscr{O}(tn\log^2 n(l+k+n)\log^2(l+k+n))$
Benaloh-Leichter	$\mathscr{O}(n^{1+\sqrt{2}}\log n(l+k+\log\log n))$

This is because the Benaloh-Leichter construction generates shares only using additions, while the Cramer-Fehr construction involves expensive multiplications.

3.3.3 Random Bits

Considering the number of random bits needed to generate a set of shares, the Cramer-Fehr construction has the advantage:

Construction	Random bits
Cramer-Fehr	$\mathscr{O}((l+k+n)t\log n)$
Benaloh-Leichter	$\mathscr{O}((l+k+\log n)n^{1+\sqrt{2}})$

We will end this discussion on LISS here. Later, we will recall this for efficiency comparison with our scheme.

4 CRT based Schemes

In this chapter, we will explore Asmuth-Bloom's secret sharing scheme, which is based on the Chinese Remainder Theorem (CRT). This scheme is of particular interest because all other CRT-based schemes [Mig83, IB05, GRS99] can be considered special cases of the Asmuth-Bloom scheme. Therefore, understanding the Asmuth-Bloom scheme [AB83] provides a comprehensive foundation for CRT-based secret-sharing techniques.

Share Distribution: The dealer selects integers m_0 and $m_1 < m_2 < \cdots < m_n$ satisfying the following conditions:

$$\forall i, j \in [n] \cup \{0\}, i \neq j \Longrightarrow \gcd(m_i, m_j) = 1,$$

$$m_0 \prod_{i=n-t+2}^{n} m_i < \prod_{i=1}^{t} m_i.$$

The dealer then chooses the secret $s \in [0, m_0 - 1]$ and randomly selects an integer α such that

$$s + \alpha m_0 \in \left(\prod_{i=n-t+2}^{n} m_i, \prod_{i=1}^{t} m_i \right).$$

The share s_i for the i-th party is calculated as:

$$s_i = (s + \alpha m_0) \bmod m_i,$$

and is sent to the i-th party privately.

Secret Reconstruction: Suppose t parties $\{i_1, \ldots, i_t\} \subseteq [n]$ want to recover the secret. They pool their shares together and solve the following system of congruences:

$$\begin{cases} x \equiv s_{i_1} & (\bmod m_{i_1}) \\ x \equiv s_{i_2} & (\bmod m_{i_2}) \\ \vdots & \\ x \equiv s_{i_t} & (\bmod m_{i_t}) \end{cases}$$

They obtain a unique solution x_0 in the range $[0, \prod_{k=1}^{t} m_{i_k} - 1]$. Since $s + \alpha m_0$ also satisfies this system of congruences and

$$s + \alpha m_0 < \prod_{i=1}^{t} m_i \leq \prod_{k=1}^{t} m_{i_k},$$

we have $s + \alpha m_0$ in the range $[0, \prod_{k=1}^{t} m_{i_k} - 1]$. By the uniqueness of the solution, $s + \alpha m_0 = x_0$, and the secret can be recovered by computing $s = x_0 \bmod m_0$.

4.1 Information Rate of Asmuth-Bloom's Scheme

Asmuth-Bloom's secret sharing scheme is a threshold scheme, where the information rate is a measure of the efficiency of the scheme. The information rate r is defined as the ratio of the size of the secret to the size of the largest share.

Let:

- s be secret, which is an integer in the range $[0, m_0 - 1]$.
- $m_0, m_1, \ldots, m_n$ be the moduli used in the scheme.
- s_i be the share given to the i-th party, calculated as $s_i = (s + \alpha m_0) \bmod m_i$.

The size of the secret is $\log_2 m_0$ bits because the secret s can take any value from 0 to $m_0 - 1$.

The size of each share s_i is $\log_2 m_i$ bits, as each share is an integer modulo m_i.

To ensure a fair comparison, we consider the size of the largest share, which is determined by the largest modulus m_i. Thus, the information rate r is defined as:

$$r = \frac{\text{size of secret}}{\text{size of the largest share}} = \frac{\log_2 m_0}{\max(\log_2 m_i)}$$

Since $\log_2$ is a logarithm to the base 2, the information rate can be rewritten as:

$$r = \frac{\log_2 m_0}{\max(\log_2 m_1, \log_2 m_2, \ldots, \log_2 m_n)}$$

Various studies have examined the properties of Asmuth-Bloom's scheme [QPV02]. It has been shown that the entropy of the secret decreases "not too much" when $t-1$ shares are known. Additionally, it is recommended to choose $m_0, m_1, \ldots, m_n$ as primes that are as close as possible. It has been proven that $t-2$ or fewer shares give no information about the secret in a (t,n) threshold scheme. Furthermore, when the moduli are consecutive primes, Asmuth-Bloom's scheme is asymptotically ideal. However, the information rate is always less than 1 for fixed moduli values, especially when the moduli are not large.

4.2 Our work CRT-based ramp scheme

As we have seen Asmuth-bloom's scheme lacks an information rate and the LISS scheme complexity depends on various factors like the number of parties etc hence inefficient. So we need a better alternative for integer secret sharing hence we propose a more efficient way for sharing integers using CRT

We propose an elegant solution to this situation by introducing a ramp model in the usual sharing schemes defined above.

Parameters

- **Parties**: $P = \{P_1, P_2, \ldots, P_n\}$ where each P_i owns an odd prime.
- **Privacy Threshold**: t, and **Reconstruction Threshold**: T such that $t < T$.
- **Secret**: secret x is chosen from $[-2^{\sigma-1}, 2^{\sigma-1}]$
- **Random Integer**: $r \in \left[\frac{-L}{2}, \frac{L}{2}\right]$, where L is a chosen integer.
- **Set notation**: $\{i_j\}_{j=1}^{t}$ and $\{i_j\}_{j=1}^{T}$ are subsets of parties with sizes t and T, respectively.

Key Definitions

- $\mathbf{P}^{\mathbf{t}}_{\text{Max}}$: Maximum product of primes from among the subsets $\{i_j\}_{j=1}^{t}$.
- $\mathbf{P}^{\mathbf{T}}_{\text{Min}}$: Minimum product of primes among subsets $\{i_j\}_{j=1}^{T}$.
- Condition: $P^{t}_{\text{Max}} < P^{T}_{\text{Min}}$.

Sharing of Secret x To securely share the secret x, a masked integer S is defined as:

$$S = x + 2^{\sigma} \cdot r$$

where r is uniformly random from the interval defined above, and σ is a fixed chosen parameter.

Procedure

1. **Secret Masking**: Compute $S = x + 2^{\sigma} \cdot r$
2. **Share Distribution**: Each party P_i computes $s_i = S \mod p_i$.

Reconstruction of Secret Authorized parties $\{i_j\}_{j=1}^{T}$ can reconstruct S using their shares $\{s_i\}_{i \in \{i_j\}_{j=1}^{T}}$ through the Chinese Remainder Theorem:

$$S = \sum_{i \in \{i_j\}_{j=1}^{T}} a_i \cdot s_i \mod \prod_{j=1}^{T} p_{i_j}$$

where a_i are integers derived from the CRT, and then compute:

$$x = S \mod (2^{\sigma})$$

4.2.1 Theorem

The secret-sharing scheme above satisfies the following:

- **Correctness**: The scheme is perfectly correct if $|S| \leq \frac{P^T_{Min}}{2}$ i.e equivalent to $(L+2) \cdot 2^{\sigma} < P^T_{\text{Min}}$.
- **Security**: The insecurity of the scheme is less than $\frac{P^t_{\text{Max}}}{L}$. That is, for any unauthorized set, the statistical distance between the distributions of its secret shares for any two distinct secrets is at most $\frac{P^t_{\text{Max}}}{L}$.

Proof of Correctness: The scheme guarantees perfect correctness under the following conditions:

$$|S| < \frac{P^T_{\text{Min}}}{2}$$

This ensures that x can be accurately reconstructed from S using modular arithmetic.

Given $|r| < \frac{L}{2}$ and $x < 2^{\sigma-1}$, the expression $S = x + 2^{\sigma} \cdot r$ ensures that:

$$|S| \leq 2^{\sigma} \cdot \left(\frac{L}{2} + 1\right)$$

If $(L+2) \cdot 2^{\sigma} < P^T_{\text{Min}}$, then the reconstruction using the Chinese Remainder Theorem (CRT) will correctly yield $x = S \mod 2^{\sigma}$.

Proof of Security: The security of the scheme relies on the unpredictability of the residue from of t-corrupted shares $\{S' \mod P^t_{\text{Max}}\}$ compared to a uniform distribution $U(P^t_{\text{Max}})$.

Security Analysis The statistical distance $\text{SD}(\{S' \mod P^t_{\text{Max}}\}, U(P^t_{\text{Max}}))$ is bounded by $\frac{P^t_{\text{Max}}}{L}$, ensuring resilience against adversarial analysis.

As earlier, we have seen the result by Cramer-Fehr that no perfect security over infinite domains and the limited size of residue classes modulo each prime make the distribution of shares not exactly uniform. Hence, there is statistical data leakage, and hence, we have the following theorem, which defines a bound on the distributions.

4.2.2 Theorem

Let $M < L$ be arbitrary integers. Let α be an arbitrary integer coprime with M. Let x be any integer. Then we have,

$$\text{SD}((x + \alpha \cdot u_L) \mod M, U(M)) < \frac{M}{L}$$

Proof: Since $M < L$, we can express L as $L = M \cdot q + r$ where $0 \leq r < M$. Define $L' = M \cdot q$. Observe that $L' = M \cdot q$. Therefore, the statistical distance:

$$\text{SD}\left((x + \alpha \cdot u_{L'}) \mod M, U(M)\right) = 0$$

We apply the data processing inequality, which states that the distance between distributions derived from $U(L)$ and $U(L')$ using modular operations is bounded by the distance between $U(L)$ and $U(L')$:

$$\mathrm{SD}\left((x+\alpha\cdot u_L) \mod M, (x+\alpha\cdot u_{L'}) \mod M\right) \leq \mathrm{SD}(U(L), U(L'))$$

Combining (1) and (2) using the triangle inequality,

$$\begin{aligned}
&\mathrm{SD}\left((x+\alpha\cdot u_L) \mod M, U(M)\right) \\
\leq &\mathrm{SD}\left((x+\alpha\cdot u_L) \mod M, (x+\alpha\cdot u_{L'}) \mod M\right) + \mathrm{SD}\left((x+\alpha\cdot u_{L'}) \mod M, U(M)\right) \\
\leq &\mathrm{SD}(U(L), U(L')) + 0 \\
< &\frac{M}{L}
\end{aligned}$$

Hence, as L increases, $\frac{M}{L}$ decreases, indicating closer statistical proximity of the given distribution to a uniform distribution.

4.3 Challenges

The scheme aims to enhance the efficiency of cryptosystems in terms of security, communication, and information. However, in achieving these goals, our scheme encounters some limitations that are typically expected to be addressed by a cryptographic tool.

1. **Non-linearity**: In our scheme, the secret x is a non-linear function of its shares. The secret x is reconstructed using the following formula:

$$x = \left(\sum_{i=1}^{n} \lambda_i \cdot s_i \mod \left(\prod_{i=1}^{n} p_i\right)\right) \mod 2^{\sigma}$$

   This non-linearity can complicate the process of secret reconstruction and introduces challenges in the scheme's practical implementation.

2. **Integer Growth**: Following non-linearity, the issue of integer growth arises when performing arithmetic operations using our scheme. For example, suppose x and y are secretly shared using our scheme. The parties hold shares of both x and y. If the parties want to compute $x+y$, they simply add their corresponding shares x_i and y_i, generated by the masked integers $X = x + 2^{\sigma}\cdot r_1$ and $Y = y + 2^{\sigma}\cdot r_2$. Even if $|X|, |Y| < \frac{P_Q}{2}$ (where P_Q is the product of primes in the qualified set Q), to reconstruct $x+y$ correctly, the condition $|X+Y| < \frac{P_Q}{2}$ must hold. Additionally, since we are working with integers, it is essential that $|x+y| < 2^{\sigma-1}$ However, this condition is not always guaranteed, leading to potential inaccuracies. A similar issue arises with scalar multiplication $c\cdot x$ and multiplication $x\cdot y$. Therefore, it is crucial to handle these computations carefully to maintain the integrity of the operations.

4.3.1 Setting up the Security Parameter

The security of the scheme relies heavily on the factor $\frac{P^t_{\mathrm{Max}}}{L}$. Therefore, the choice of L plays a crucial role: As we have seen from theorem 4.2.2 larger the value of L, the closer the adversarial view is to a uniform distribution. In this section, we will introduce some notations for future convenience.

Let λ denote the statistical-security parameter, and let $p_1, p_2, \ldots, p_N$ represent k-bit primes.

Given $S = x + 2^{\sigma}\cdot r_L$, where $-2^{\sigma-1} < x < 2^{\sigma-1}$ and $r_L \in \left[-\frac{L}{2}, \frac{L}{2}\right]$, we consider the worst-case scenario:

$$\frac{P^t_{\mathrm{Max}}}{2} < |S| < \frac{P^T_{\mathrm{Min}}}{2}$$

This can be simplified to:

$$\frac{P^t_{\mathrm{Max}}}{2} < 2^{\sigma}\left(\frac{L}{2}+1\right) < \frac{P^T_{\mathrm{Min}}}{2}$$

Therefore, we aim to select L such that $|S|$ falls within the range:

$$P_{\text{Max}}^{t} < L < P_{\text{Min}}^{T}$$

This implies:

$$2^{k \cdot t} < L < 2^{k \cdot T}$$

A suitable choice for L can be approximated as:

$$L \approx 2^{k \cdot t + \lambda}$$

where λ is the statistical-security parameter.

5 Multi-party Protocol

Secret-sharing schemes are fundamental tools used to develop secure MPC protocols. Shamir's secret-sharing scheme is commonly employed due to its ideal security properties. In this section, we derive the BGW MPC framework using our CRT-based scheme within the semi-honest corruption model and an honest majority setting.

5.0.1 Notations

All the notations from the secret-sharing scheme are preserved. Additionally, we denote the sharing of some input x as

$$\text{Share}(P,t,T,x) \Longrightarrow \{[x]_i\}_{i\in[n]}$$

where $[n] = \{1,2,3,\ldots,n\}$ and $P = \prod_{i\in[n]} p_i$. Masking is denoted by

$$\text{msk}(x) = S = x + 2^{\sigma}\cdot r$$

and reconstruction is given by

$$\text{Reconstruct}(\{x_i\}_{i\in[n]}) = S.$$

5.0.2 Overview

We address the integer growth problem discussed previously. For convenience, let σ be of size λ. For every input x, where the parameter $L = 2^{k.t+\lambda}$, the masked value $\text{msk}(x)$ is at most of size

$$2^{k.t+\lambda}\cdot 2^{\sigma} \leq 2^{k.t+2\lambda}.$$

For each input x, we secret share the value x using our scheme where the parameter L is $2^{k.t+\lambda}$. Therefore,

$$\text{msk}(s) \text{ is of size at most } (2^{k.t+\lambda}+2)\cdot 2^{\sigma} \leq 2^{k.t+2\lambda}.$$

Throughout the MPC protocol, we maintain the invariant that for input x, the secret integer $S = \text{msk}(s)$ associated with the secret share of x is upper-bounded by some $\text{poly}(\lambda)\cdot 2^{k.t+2\lambda}$.

This bound is maintained for each addition gate if the circuit is of $poly(\lambda)$ size. However, after each multiplication gate (including scalar multiplication where the scalar is superpolynomial in λ), this invariant is broken. Hence, we employ a degree reduction protocol to re-establish this invariant.

5.0.3 Degree Reduction

The degree reduction protocol reduces the size of the mask without altering the secret it encodes, ensuring that the masked integer carrying the secret falls within the reconstruction range. This protocol is typically employed after a multiplication gate to handle the growth of the integer. In the preprocessing phase, every party generates two secret shares of a random value r, denoted by $[r]^0$ and $[r]^1$. The instance $[r]^0$ is sampled with parameter $L = 2^{kt+\lambda}$, while $[r]^1$ is sampled with parameter $L = 2^{2kt+5\lambda}$. This choice of L is necessary because the maximum size of $\text{msk}(x\cdot y)$ is $2^{2kt+4\lambda}$.

Parties use $[r]^1$ as a mask to reconstruct the value $r + x\cdot y$ in the clear and then deduct the secret share $[r]^0$ locally. To successfully reconstruct the value $r + x\cdot y$, which corresponds to an integer of size at most $2^{2kt+5\lambda}$, it is crucial that $\text{msk}(r + x\cdot y) < P$. If this condition is not met, the reconstruction fails, rendering the degree reduction incorrect. Therefore, ensuring the masked value stays within this range is essential for the protocol's correctness.

5.1 Sub-protocols

5.1.1 Randomness Generation(F_{rand})

Generating shares of random value F_{rand} In this sub-protocol, parties generate a secret sharing of a random value to threshold parameter L. Communication cost per party is $\mathbf{O}(\sum_{\mathbf{i}} \log(\mathbf{p_i}))$

- For all $i \in [n]$, the i^{th} party samples a random value $r_i \in [-2^{\sigma-1}, 2^{\sigma-1}]$ Secret shares r_i i.e $Share(P,t,T,r_i) \Longrightarrow \{[r_i]_j\}_{j\in[n]}$ and send shares to the parties.
- For all $i \in [n]$, the i-th party locally compute $[r]_i = \sum_{j\in[n]}[r_j]_i \mod p_i$ which in-turn secret share of the random Z-element $r = \sum_{j\in[n]} r_j$

5.1.2 Random Double sharing protocol

In this protocol, parties obtain shares of the same value but sharing is obtained in different threshold parameters.Communication cost per party is $\mathbf{O}\left(\sum_{\mathbf{i}} \log(\mathbf{p_i})\right)$

Each $P_i \in \mathscr{P}$ given the threshold parameters $L = 2^{k.t+\lambda}$ and $L' = 2^{2kt+5\lambda}$ Compute the following,

$$\left([r]_1^0, \ldots, [r]_n^0\right) \leftarrow F_{rand}(L)$$
$$\left([r]_1^1, \ldots, [r]_n^1\right) \leftarrow F_{rand}(L' | \text{with same } r_i \text{ parties sampled while parameter is L})$$

and output $\left([r]_i^0, [r]_i^1\right)$ as double share of r for party i

5.1.3 Degree reduction protocol $F_{\mathbf{deg}}$

In this sub-protocol, parties re-sample the secret share of some x such that the corresponding integer Mask of x is small enough for perfect reconstruction. Communication cost per party is $\mathbf{O}(\log(\mathbf{p_i}))$

- Input. Parties hold the secret shares $[x]$ of some input x.
- Additionally, parties run a Random Double sharing protocol and obtain a fresh double-shared sample.
- Parties broadcast their share of $x+r$.
- Given all the secret shares, parties locally reconstruct $x+r \in \mathbb{F}$ and subtract $\left[[r]_i^0\right]_{i\in[n]}$ from $\{(x+r) \mod p_i\}$

5.2 Protocol

Using the above subprotocols, we are now ready to provide the final main and full BGW protocol based on our scheme. Let M denote the circuit's multiplicative depth, which is the number of multiplication gates, including scalar multiplication.

♦ Preprocessing Phase

- Each party $k \in [n]$ individually generates a sharing of a random element r^i of his choice from the secret space as $\text{Share}(P,t,T,r^i) \rightarrow \{[r^i]_k\}_{k\in[n]}$ and sends the shares to all other parties.
- Parties generate M double sharing samples $[r]^0, [r]^1$ using F_{rand} and the double sharing protocol described above, preserving the mentioned threshold parameters.
- Parties generate secret sharing $[0]_{i\in[n]}$ of 0 as described in the F_{rand} protocol.

♦ Protocol Run

- Parties share their secret $x^i \in$ Secret space by broadcasting $x^i + r^i$, where r^i is the party's internal randomness as described in step 1 of the preprocessing phase.
- To obtain shares of $[S_i]$:
  - Each party k performs $(x^i + r^i) - r^i_k \mod p_k$, where r^i_k is party k's share of r^i obtained from the preprocessing step.
- Addition gate $(x^i + x^j)$ is realized by parties locally computing the addition of shares $[S_i]_k$ and $[S_j]_k$.
- Multiplication gate $(x^i \cdot x^j)$ is realized by the following steps:
  - Each party $k \in [n]$ locally multiplies shares $[S_i]_k \cdot [S_j]_k$.
  - Apply the degree reduction F_{deg} protocol to the multiplied share vector and obtain a new sharing of $x^i \cdot x^j$.
- Scalar Multiplication gate $(c \cdot x^i)$:
  - Parties locally multiply c with the share of S_i they have.
  - Apply the degree reduction F_{deg} protocol to the vector and obtain a new sharing of $c \cdot x^i$.

♦ Reconstruction of Output

- Parties hold secret shares of $[out]$ and also hold a secret sharing of 0 generated in the preprocessing phase.
- Each party P_k locally computes and broadcasts $([0]_k + [\text{out}]_k) \mod p_k$ as the secret shares of $0 + \text{out}$.
- Parties broadcast their shares.
- Parties locally reconstruct $0 + \text{out}$ to obtain the value of the output.

5.3 Security of the protocol

In this section, we will prove the protocol's semi-honest security. We will show that the protocol is statistically secure against semi-honest adversaries, as the underlying scheme is statistically secure.

5.3.1 Theorem

Let C be an arithmetic circuit defined over $\mathbb{Z}$. We say that given the above subprotocols, the real-world protocol π securely realizes C with corruption threshold t and reconstruction threshold T if for any n-party input vector $\vec{x}$ and any subset $I \subset [n]$ such that $|I| \leq t$, there exists an efficient simulator S such that

$$SD(((S(I,\vec{x}_I,C(\vec{x})_I)),C(\vec{x})),(view_I^{\pi}(\vec{x}),Output^{\pi}(\vec{x}))) \leq \varepsilon(\lambda),$$

where SD denotes the statistical distance.

Proof. We will prove that the MPC protocol described above is semi-honest secure. Let A denote the real-world adversary and S denote the simulator. The simulation works as follows:

Preprocessing

The simulator S simulates the preprocessing protocols honestly. Since this step involves randomness generation, there is no simulation error in this step. The simulator's generated randomness will be identical to the real protocol's randomness, ensuring no distinguishable difference from the adversary's perspective.

Sharing the Inputs

For all corrupt parties, the simulator freshly shares their inputs as per the protocol description. For the honest parties, the simulator simply chooses 0 as their secrets. Because the protocol masks these secrets with random values, the resulting shares appear random, similar to the real protocol. Therefore, the simulated view will be $2^{-\lambda}$ close to the real view due to the security of the secret sharing scheme.

Addition/Subtraction

The simulator locally computes the addition and subtraction of secret shares for the adversary. These operations do not alter the shares themselves but merely combine them. Thus, there is no difference between the simulated and real views, leading to no simulation error.

Multiplication

The simulator computes the secret shares $[x]_i \cdot [y]_i + [r]_i^1$ locally. To correctly simulate the opening of $x \cdot y + r$, additional randomness must be simulated as $r = \sum_{i=1}^{n} r_i$.

For all $i \in A$, the shares are:

$$[x]_i \cdot [y]_i + [r]_i^1 = [x]_i \cdot [y]_i + \sum_{i \in I} [r_i]_i + \sum_{j \notin I} [r_j]_i.$$

The honest parties' randomness is sampled to satisfy this relation. Specifically, the simulator generates random values for honest parties ensuring:

$$\sum_{j \notin I} [r_j]_i = [r]_i^1 - \sum_{i \in I} [r_i]_i.$$

Once this randomness is chosen, the simulator can open the value $x \cdot y + r$ by preserving the adversarial part and masking it with the randomness generated for the honest parties. The threshold parameter $\mathbf{L}' = \mathbf{2kt} + \mathbf{5}\lambda$ ensures the masking is secure. The simulator then reveals and subtracts the share of $[r]_i^0$ from the resulting value.

Output Reconstruction

The simulator locally computes $[out]_i + [0]_i$ for the adversarial part, where $[0]_i$ denotes the zero shares of the honest parties. The simulation here is similar to the multiplication gate, ensuring that the adversary's view in the output reconstruction phase matches the real protocol's view within a negligible statistical distance.

Thus, for any input vector $\vec{x}$ and any subset $I \subset [n]$ such that $|I| \leq t$, the simulated view by S is statistically close to the real view, with the statistical distance bounded by a negligible function $\varepsilon(\lambda)$. Hence, the protocol π securely realizes the arithmetic circuit C under the given corruption threshold t and reconstruction threshold T.

□

6 Complexity and Efficiency

In this section, we discuss the complexity and efficiency of our secret-sharing scheme and compare it with other protocols in the literature. We will analyze the computational complexity of different phases of the protocol and compare its performance with two well-known schemes: LISS and Asmuth-Bloom CRT-based schemes.

6.0.1 Complexity of the Secret Sharing Scheme

The complexity of our secret sharing scheme can be broken down into three main phases:

- **Sharing Phase**: This phase requires 1 round of communication with a complexity of $O(\sum_i \log(p_i))$, where p_i denotes the prime modulus used in the scheme.
- **Reconstruction Phase**: This phase also requires 1 round of communication with a complexity of $O(n \cdot \log(p_i))$ per party, where n is the number of parties.

6.0.2 Complexity of the MPC Protocol per Party

The complexity of the multi-party computation (MPC) protocol per party is analyzed as follows:

- **Preprocessing Phase**: The communication required in this phase is $O(\sum_i \log(p_i))$.
- **Protocol Run Phase**: During this phase, the communication required is $O(\max\{\sigma, \log(p_i)\})$, where σ is a parameter defined in the scheme.
- **Reconstruction Phase**: The communication required in this phase is $O(\log(p_i))$.

6.0.3 Efficiency of the Scheme

The efficiency of our scheme can be evaluated based on the share size and the information rate. The share size for each party is $O(\log(p_i))$, and the secret size is $\log(\sigma)$. Therefore, the information rate of the scheme is given by:

$$r = \frac{\sigma}{\max(\log p_1, \log p_2, \ldots, \log p_n)}$$

This information rate highlights the efficiency of our scheme, which is competitive compared to other approaches.

6.1 Comparison with Existing Schemes

We now compare our scheme against two established schemes: LISS and Asmuth-Bloom.

6.1.1 Comparison with the LISS Scheme

One key aspect of our scheme's advantage over the LISS scheme is the share size. For a threshold-t scheme $T_{t,n}$ with a secret of bit length l and a statistical security parameter k, the share sizes for different constructions are:

Construction	Share Size
Cramer-Fehr	$\mathcal{O}((l+k+n)\log n)$
Benaloh-Leichter	$\mathcal{O}((l+k+\log\log n)n^{\sqrt{2}})$

Table 1: Share Sizes for Different Constructions

In contrast, the share size in our scheme is $O(\log(p_i))$, which is independent of both the circuit size and the number of parties. This is a significant improvement over the LISS scheme, where the share size grows with the number of parties and the complexity of the secret sharing scheme.

Random Bits When it comes to the number of random bits required, the Cramer-Fehr construction has the following complexity:

Construction	Random Bits
Cramer-Fehr	$\mathcal{O}((l+k+n)t\log n)$
Benaloh-Leichter	$\mathcal{O}((l+k+\log n)n^{1+\sqrt{2}})$

Table 2: Random Bits Required for Different Constructions

Our scheme requires random bits only at the initial step for masking, which involves $O(kt+\lambda)$. This requirement is much smaller compared to the Benaloh-Leichter scheme, which requires a large number of random bits due to the complexity of the randomization process.

6.1.2 Information Rate Comparison

Our scheme also compares favorably against the Asmuth-Bloom CRT-based scheme in terms of information rate. The information rate for the Asmuth-Bloom scheme is given by:

$$r = \frac{\log m_0}{\max(\log m_1, \log m_2, \ldots, \log m_n)}$$

In the Asmuth-Bloom scheme, the stringent conditions on the moduli lead to a low information rate. Specifically, the conditions for the moduli are:

$$\forall i, j \in [n] \cup \{0\}, i \neq j \Longrightarrow \gcd(m_i, m_j) = 1,$$

$$m_0 \prod_{i=n-t+2}^{n} m_i < \prod_{i=1}^{t} m_i.$$

These conditions necessitate large moduli, which in turn result in a low information rate.

In contrast, our scheme uses a relaxed threshold condition through a ramp setting, allowing us to choose primes of sufficient size while maintaining a high information rate. In an ideal case where primes are around the size of σ, our scheme achieves an optimal information rate similar to that of Shamir's scheme.

6.2 Conclusion

Overall, our scheme demonstrates improved efficiency and complexity compared to existing protocols. Its smaller share sizes, reduced random bit requirements, and better information rate position it as a competitive alternative to established schemes like LISS and Benaloh-Leichter, and offer advantages over the Asmuth-Bloom CRT-based scheme in terms of information rate and practical applicability.

References

[AB83] Charles Asmuth and John Bloom. A modular approach to key safeguarding. *IEEE Trans. Inf. Theory*, 29(2):208–210, 1983.

[ABF+19] Benny Applebaum, Amos Beimel, Oriol Farràs, Oded Nir, and Naty Peter. Secret-sharing schemes for general and uniform access structures. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of LNCS, pages 441–471. Springer, Heidelberg, May 2019.

[ABNP20] Benny Applebaum, Amos Beimel, Oded Nir, and Naty Peter. Better secret sharing via robust conditional disclosure of secrets. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *52nd ACM STOC*, pages 280–293. ACM Press, June 2020.

[AL17] Gilad Asharov and Yehuda Lindell. A full proof of the BGW protocol for perfectly secure multiparty computation. *Journal of Cryptology*, 30(1):58–151, January 2017.

[Bea92] Donald Beaver. Efficient multiparty protocols using circuit randomization. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of LNCS, pages 420–432. Springer, Heidelberg, August 1992.

[Bea00] Donald Beaver. Minimal-latency secure function evaluation. In *EUROCRYPT*, pages 335–350, 2000.

[BB89] J. Bar-Ilan and D. Beaver. Non-cryptographic fault-tolerant computing in a constant number of rounds of interaction. In *PODC*, pages 201–209, 1989.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

[BM84] G. R. Blakley and Catherine Meadows. Security of ramp schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of LNCS, pages 242–268. Springer, Heidelberg, August 1984.

[BL88] J. C. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In S. Goldwasser, editor, *CRYPTO*, volume 403 of Lecture Notes in Computer Science, pages 27–35. Springer, 1988.

[BI01] Amos Beimel and Yuval Ishai. On the power of nonlinear secret-sharing. *IACR Cryptol. ePrint Arch.*, page 30, 2001.

[Chor85] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *FOCS*, pages 383–395. IEEE, 1985.

[Chor93] B. Chor and E. Kushilevitz. Secret sharing over infinite domains. *J. Cryptology*, 6(2):87–95, 1993.

[Chor95] B. Chor, M. Geréb-Graus, and E. Kushilevitz. Private computations over the integers. *SIAM J. Comput.*, 24(2):376–386, 1995.

[Can96] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multiparty computation. In *STOC*, pages 639–648, 1996.

[Can00] R. Canetti. Security and Composition of Multiparty Cryptographic Protocols. In the Journal of Cryptology, 13(1):143–202, 2000.

[CF00] R. Cramer and S. Fehr. Optimal black-box secret sharing over arbitrary abelian groups. In Yung [91], pages 272–287, 2000.

[CDM00] R. Cramer, I. Damgård, and U. M. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *EUROCRYPT*, pages 316–334, 2000.

[Des88] Yvo Desmedt. Society and group oriented cryptography: A new concept. In Carl Pomerance, editor, *CRYPTO'87*, volume 293 of LNCS, pages 120–127. Springer, Heidelberg, August 1988.

[DF90] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of LNCS, pages 307–315. Springer, Heidelberg, August 1990.

[DF94] Y. Desmedt and Y. Frankel. Perfect homomorphic zero-knowledge threshold schemes over any finite abelian group. *SIAM J. Discrete Math.*, 7(4):667–679, 1994.

[DIK+08] Ivan Damgård, Yuval Ishai, Mikkel Krøigaard, Jesper Buus Nielsen, and Adam Smith. Scalable multiparty computation with nearly optimal work and resilience. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of LNCS, pages 241–261. Springer, Heidelberg, August 2008.

[DIK10] Ivan Damgård, Yuval Ishai, and Mikkel Krøigaard. Perfectly secure multiparty computation and the computational overhead of cryptography. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of LNCS, pages 445–465. Springer, Heidelberg, May / June 2010.

[DN07] Ivan Damgård and Jesper Buus Nielsen. Scalable and unconditionally secure multiparty computation. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of LNCS, pages 572–590. Springer, Heidelberg, August 2007.

[DT06] I. Damgård and R. Thorbek. Linear integer secret sharing and distributed exponentiation. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *Public Key Cryptography*, volume 3958 of Lecture Notes in Computer Science, pages 75–90. Springer, 2006.

[EPKP+20] Oğuzhan Ersoy, Thomas Brochmann Pedersen, Kamer Kaya, Ali Aydın Selçuk, and Emin Anarim. A CRT-Based Verifiable Secret Sharing Scheme Secure Against Unbounded Adversaries. *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1245–1262. ACM, 2020.

[Fitzi97] M. Fitzi, M. Hirt, and U. M. Maurer. Trading correctness for privacy in unconditional multiparty computation (extended abstract). In *CRYPTO*, pages 121–136, 1997.

[FY92] Matthew K. Franklin and Moti Yung. Communication complexity of secure computation (extended abstract). In *24th ACM STOC*, pages 699–710. ACM Press, May 1992.

[Genn98] R. Gennaro, M. O. Rabin, and T. Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In *PODC*, pages 101–111, 1998.

[GRS99] Oded Goldreich, Dana Ron, and Madhu Sudan. Chinese remaindering with errors. In *31st ACM STOC*, pages 225–234. ACM Press, May 1999.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[Hirt97] M. Hirt and U. M. Maurer. Complete characterization of adversaries tolerable in secure multiparty computation (extended abstract). In *PODC*, pages 25–34, 1997.

[IK00] Y. Ishai and E. Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *FOCS*, pages 294–304, 2000.

[IB05] Sorin Iftene and Ioana Boureanu. Weighted threshold secret sharing based on the Chinese remainder theorem. *Sci. Ann. Cuza Univ.*, 15:161–172, 2005.

[LV18] Tianren Liu and Vinod Vaikuntanathan. Breaking the circuit-size barrier in secret sharing. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th ACM STOC*, pages 699–708. ACM Press, June 2018.

[LSP82] L. Lamport, R. Shostack, and M. Pease. The Byzantine Generals Problem. In *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[Mig83] Maurice Mignotte. How to share a secret? In Thomas Beth, editor, *EUROCRYPT'82*, volume 149 of LNCS, pages 371–375. Springer, Heidelberg, March / April 1983.

[NIST18] National Institute of Standards and Technology. Multi-party threshold cryptography, 2018–present.

[Ost91] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks (extended abstract). In *PODC*, pages 51–59, 1991.

[QPV02] M. Quisquater, B. Preneel, and J. Vandewalle. On the security of the threshold scheme based on the Chinese remainder theorem. In D. Naccache and P. Paillier, editors, *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002*, volume 2274 of Lecture Notes in Computer Science, pages 199–210. Springer-Verlag, 2002.

[Sha79] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.

[SC17] C. Stathakopoulous and Christian Cachin. Threshold signatures for blockchain systems. *Swiss Federal Institute of Technology*, 30, 2017.

[SW99] Douglas R. Stinson and Ruizhong Wei. An application of ramp schemes to broadcast encryption. *Inf. Process. Lett.*, 69(3):131–135, 1999.

[VNS+03] V. Vinod, Arvind Narayanan, K. Srinathan, C. Pandu Rangan, and Kwangjo Kim. On the power of computational secret sharing. In Thomas Johansson and Subhamoy Maitra, editors, *INDOCRYPT 2003*, volume 2904 of LNCS, pages 162–176. Springer, Heidelberg, December 2003.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.

[Yao82] A. C. Yao. Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS '08. 23rd Annual Symposium on*, pages 160–164, 1982.

Thank you