Commitments with Efficient Zero-Knowledge Arguments from Subset Sum Problems

Jules Maire and Damien Vergnaud

Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

Abstract. We present a cryptographic string commitment scheme that is computationally hiding and binding based on (modular) subset sum problems. It is believed that these NP-complete problems provide post-quantum security contrary to the number theory assumptions currently used in cryptography. Using techniques recently introduced by Feneuil, Maire, Rivain and Vergnaud, this simple commitment scheme enables an efficient zero-knowledge proof of knowledge for committed values as well as proofs showing Boolean relations amongst the committed bits. In particular, one can prove that committed bits $m_0, m_1, ..., m_\ell$ satisfy $m_0 = C(m_1, ..., m_\ell)$ for any Boolean circuit C (without revealing any information on those bits). The proof system achieves good communication and computational complexity since for a security parameter λ, the protocol's communication complexity is $\tilde{O}(|C|\lambda + \lambda^2)$ (compared to $\tilde{O}(|C|\lambda^2)$ for the best code-based protocol due to Jain, Krenn, Pietrzak and Tentes).

1 Introduction

A commitment scheme [7] is a cryptographic protocol that enables one party to commit to a value (or set of values) without revealing it, while ensuring that this value cannot be modified. In constructing sophisticated cryptographic protocols, it can be necessary to prove some property of a committed message without revealing anything more than the property itself. This is usually achieved through the use of zero-knowledge proofs of knowledge [14]. This *commit-and-prove* paradigm [20,11] is used in many areas of applied cryptography (anonymous credentials, blockchains, electronic voting, ...).

In 1994, Shor [26] introduced a quantum algorithm that could break cryptosystems based on the hardness of factoring large integers or solving discrete logarithm problems. This has emphasized the need for new cryptographic systems, leading to the emergence of a new field, known as *post-quantum cryptography*, which focuses on creating cryptographic algorithms that are secure against quantum (and classical) computers.

The (modular) subset sum problem is to find, given integers t and q, a subset of given integers $\gamma_1, \ldots, \gamma_n$, whose sum is t modulo q. This NP-complete problem was used in the 1980s, following [22], for the construction of several public-key encryption schemes. The majority of those schemes were broken using lattice-based techniques (see [23]), but the problem itself remains unsolvable for specific parameters and is even thought to be intractable for quantum computers.

A plethora of cryptographic constructions have been proposed whose security is based on the difficulty of the subset sum problem [15,1,21]. In a celebrated paper, Impagliazzo and Naor [15] presented in particular a pseudo-random generator and an elegant bit commitment scheme. We extend the latter to a simple string commitment scheme and provide efficient zero-knowledge proofs for any relation amongst committed values using the recent zero-knowledge proof system proposed by Feneuil, Maire, Rivain, and Vergnaud and based on the *MPC-in-the-head* paradigm.

Contributions of the paper

Commitment scheme. We first present a modified version of the bit-commitment based on the subset sum problem proposed in [15]. This new scheme enables commitments to bit-strings and is related to the one from [15] in a similar manner to how the well-known Pedersen commitment scheme [24] is related to preliminary discrete-logarithm based bit-commitments from [10,9].

The design principle is simple but seems to have been overlooked for more than 30 years (even if similar ideas have been used in lattice-based cryptography). For a security level $\lambda \in \mathbb{N}$ (*i.e.* against an adversary making 2^λ bit-operations using a $2^{\lambda/2}$-bits memory), it enables to commit to bit-strings of length $\ell \leq 2\lambda$ using a 2λ-bits modulus q and $(\ell + 2\lambda)$ integers smaller than q. The setup thus requires $O(\lambda^2)$ random or pseudo-random bits that can be generated easily using a so-called *extendable-output function* (XOF). A commitment is a sum of a (randomized) subset of these integers modulo q; therefore, it is of optimal bit-length 2λ and can be computed in $O(\lambda^2)$ binary operations. The *hiding* property (*i.e.* one cannot learn anything about the committed message from the commitment) relies on the hardness of the subset-sum problem, while its *binding* property (*i.e.* one cannot open a commitment to two different messages) relies on the hardness of the related *weighted knapsack problem*. With the proposed parameters, both problems are believed to be resistant to a quantum adversary that makes at most $2^{\lambda/2}$ qubits operations.

Zero-Knowledge Protocols. Very recently, Feneuil, Maire, Rivain, and Vergnaud [13] proposed zero-knowledge arguments for the subset sum problem. They introduced the idea of *artificial abort* to the so-called *MPC-in-the-head paradigm* [16] and achieved an asymptotic improvement by producing arguments of size $O(\lambda^2)$. Their protocol readily gives a way to prove knowledge of the committed bit-string without revealing anything about it.

We extend their work to prove that a committed triple $(b_1, b_2, b_3) \in \{0,1\}^3$ satisfy a Boolean relation (*e.g.* $b_1 \wedge b_2 = b_3$ or $b_1 \oplus b_2 = b_3$) without revealing any additional information about them. The bits can be in arbitrary positions in the same or in different commitments and the proof of the Boolean relation does not add any overhead compared to the basic opening proof. This flexibility allows proving that committed bits $m_0, m_1, ..., m_\ell$ satisfy $m_0 = C(m_1, ..., m_\ell)$ for any Boolean circuit C with good communication and computational complexity. Indeed, by packing the commitments of bits on the circuit wires, we

obtain a protocol with communication complexity $\tilde{O}(|C|\lambda + \lambda^2)$ where $|C|$ denotes the number of AND/XOR gates of C. This has to be compared with the code-based protocol due to Jain, Krenn, Pietrzak, and Tentes [17]. They provide a commitment scheme with zero-knowledge proofs from the LPN-assumption (or hardness of decoding a random linear code). This scheme has $\tilde{O}(|C|\lambda^2)$ communication complexity and allows only proving Boolean relations bit-wise on binary strings (which may result in a large overhead depending on the circuit considered). There also exist lattice-based constructions of commitment schemes with zero-knowledge proofs [6,3,2] but the messages committed are small integers. They can be used to prove the satisfiability of arithmetic circuits but proving the satisfiability of a Boolean circuit with these schemes results in an important overhead in communication and computation.

2 Preliminaries

2.1 Notations

All logarithms are in base 2. We denote the security parameter by λ, which is given to all algorithms in the unary form 1^λ. Unless otherwise stated, algorithms are randomized, and "PPT" stands for "probabilistic polynomial-time" in the security parameter. Random sampling from a finite set X according to the uniform distribution is denoted by $x \xleftarrow{\$} X$. The symbol $\xleftarrow{\$}$ is also used for assignments from randomized algorithms, and the symbol $\leftarrow$ is used for assignments from deterministic algorithms and calculations. For the sake of simplicity, we denote the set of integers $\{1, \ldots, N\}$ by $[1, N]$.

We denote integer vectors in bold print. A vector composed only of 1's or 0's is denoted as $\mathbf{1}$ or $\mathbf{0}$ respectively (its length will be clear within the context). Given two integer vectors of the same length $\boldsymbol{\gamma}$ and $\boldsymbol{x}$, $\langle \boldsymbol{\gamma}, \boldsymbol{x} \rangle$ denotes their inner-product. For two bit-strings $\boldsymbol{x} \in \{0,1\}^n$ and $\boldsymbol{y} \in \{0,1\}^m$, $(\boldsymbol{x} \| \boldsymbol{y}) \in \{0,1\}^{n+m}$ denotes the concatenation of $\boldsymbol{x}$ and $\boldsymbol{y}$, and $\boldsymbol{x} \cdot \boldsymbol{y}$ denotes the component-wise product.

Two distributions $\{D_\lambda\}_\lambda$ and $\{\tilde{D}_\lambda\}_\lambda$ are deemed (t, ϵ)-indistinguishable if, for any algorithm $\mathcal{A}$ running in time at most $t(\lambda)$, we have

$$|\Pr[\mathcal{A}(1^\lambda, x) = 1 \mid x \xleftarrow{\$} D_\lambda] - \Pr[\mathcal{A}(1^\lambda, x) = 1 \mid x \xleftarrow{\$} \tilde{D}_\lambda]| \leq \epsilon(\lambda).$$

A (ℓ, t, ϵ)-*pseudo-random generator* (PRG) is a deterministic algorithm G that, for all $\lambda \in \mathbb{N}$, on input a bit-string $\boldsymbol{x} \in \{0,1\}^\lambda$ outputs $G(\boldsymbol{x}) \in \{0,1\}^{\ell(\lambda)}$ with $\ell(\lambda) > \lambda$ such that the distributions $\{G(\boldsymbol{x}) \mid x \xleftarrow{\$} \{0,1\}^\lambda\}_\lambda$ and $\{\boldsymbol{r} \mid \boldsymbol{r} \xleftarrow{\$} \{0,1\}^{\ell(\lambda)}\}_\lambda$ are (t, ϵ)-indistinguishable. From such a generator, with $\ell(\lambda) = 2\lambda$, it is possible to construct a *tree PRG* [19], which takes a root $\boldsymbol{x} \in \{0,1\}^\lambda$ as input and generates $N = 2^t$ pseudo-random λ-bit strings in a structured fashion as follows: $\boldsymbol{x}$ is the label of the root of a depth-t complete binary tree in which the right/left child of each node is labeled with the λ most/least significant bits of the output of the PRG applied to the root label. This structure allows revealing

$N-1$ pseudo-random values of the leaves by revealing only $\log(N)$ labels of the tree (by revealing the labels on the siblings of the paths from the root to this leaf).

2.2 Commitments

Definition 1. *(Commitment scheme). A commitment scheme is a triple of PPT algorithms* $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Ver})$ *such that:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$*. On input* λ*, the* setup *algorithm outputs the public parameters* pp *containing a description of the message space* $\mathcal{M}$*.*
- $\mathsf{Com}(\mathsf{pp}, m) \to (c, \mathsf{aux})$*. On input* pp *and* $m \in \mathcal{M}$*, the* commit *algorithm outputs a commitment-opening pair* (c, aux)*.*
- $\mathsf{Ver}(\mathsf{pp}, m, c, \mathsf{aux}) \to b \in \{0,1\}$*. On input* pp*,* $m \in \mathcal{M}$ *and* (c, aux)*, the* verifying *(or* opening*) algorithm outputs a bit* $b \in \{0,1\}$*.*

Moreover, it satisfies the following correctness property: we have for all $\lambda \in \mathbb{N}$*,*

$$\Pr[\mathsf{Ver}(\mathsf{pp}, m, c, \mathsf{aux}) = 1 \mid \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda), m \xleftarrow{\$} \mathcal{M}, (c, \mathsf{aux}) \xleftarrow{\$} \mathsf{Com}(\mathsf{pp}, m)] = 1.$$

There are two security notions underlying a commitment scheme.

Definition 2. *Let* $t : \mathbb{N} \to \mathbb{N}$ *and* $\epsilon : \mathbb{N} \to [0,1]$*. A commitment scheme* $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Ver})$ *is said:*

- (t, ϵ)-computationally hiding *if for all two-phases algorithm* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, we have for all* $\lambda \in \mathbb{N}$*:*

$$\Pr\left[b = b' \;\middle|\; \begin{array}{l} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda), (m_0, m_1, s) \xleftarrow{\$} \mathcal{A}_1(\mathsf{pp}), b \xleftarrow{\$} \{0,1\} \\ (c, \mathsf{aux}) \xleftarrow{\$} \mathsf{Com}(\mathsf{pp}, m_b), b' \xleftarrow{\$} \mathcal{A}_2(c, s) \end{array}\right] \leq \frac{1}{2} + \epsilon(\lambda)$$

  when $\mathcal{A}$ *runs in time at most* $t(\lambda)$ *in this probabilistic computational game.*
- (t, ϵ)-computationally binding *if for all algorithm* $\mathcal{A}$*, we have for all* $\lambda \in \mathbb{N}$*:*

$$\Pr\left[\begin{array}{c} m_1 \neq m_2 \\ \mathsf{Ver}(pp, m_1, c, aux_1) = 1 \\ \mathsf{Ver}(pp, m_2, c, aux_2) = 1 \end{array} \;\middle|\; \begin{array}{l} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda), \\ (m_1, m_2, \mathsf{aux}_1, \mathsf{aux}_2, c) \xleftarrow{\$} \mathcal{A}(1^\lambda, \mathsf{pp})\} \end{array}\right] \leq \epsilon(\lambda)$$

  when $\mathcal{A}$ *runs in time at most* $t(\lambda)$ *in this probabilistic computational game.*

2.3 Zero-knowledge Arguments

A zero-knowledge protocol for a polynomial-time decidable binary relation $\mathcal{R}$ is defined by two interactive algorithms, a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. Both algorithms are given a common input x, and $\mathcal{P}$ is given an additional *witness* w such that $(x, w) \in \mathcal{R}$. The two algorithms then exchange messages until $\mathcal{V}$ outputs a bit b, with $b = 1$ meaning $\mathcal{V}$ accepts $\mathcal{P}$'s claim and $b = 0$ meaning $\mathcal{V}$ rejects it.

This sequence of messages and the answer b is referred to as a *transcript* and denoted $\mathrm{View}(\mathcal{P}(x,w),\tilde{\mathcal{V}}(x))$. In this paper, we consider *zero-knowledge argument of knowledge* which are protocols that allow a PPT prover to convince a PPT verifier that they *know* a witness w. There are three security notions underlying a zero-knowledge argument of knowledge.

Definition 3. *Let $t:\mathbb{N}\to\mathbb{N}$, $\epsilon,\alpha,\zeta:\mathbb{N}\to[0,1]$, and $\mathcal{R}$ be a polynomial-time decidable binary relation. A zero-knowledge argument $(\mathcal{P},\mathcal{V})$ for $\mathcal{R}$ achieves:*

- α-completeness*, if for all $\lambda\in\mathbb{N}$ and all $(x,w)\in\mathcal{R}$, with $x\in\{0,1\}^\lambda$, $\Pr[\mathrm{View}(\mathcal{P}(x,w),\mathcal{V}(x))=1]\geq 1-\alpha(\lambda)$ (*i.e. *$\mathcal{P}$ *succeeds in convincing $\mathcal{V}$, except with probability α).*
- ϵ-(special) soundness*, if for all PPT algorithm $\tilde{\mathcal{P}}$ such that for all $\lambda\in\mathbb{N}$ and all $x\in\{0,1\}^\lambda$, $\tilde{\epsilon}(\lambda):=\ Pr\left[\mathrm{View}(\tilde{\mathcal{P}}(x),\mathcal{V}(x))=1\right]>\epsilon(\lambda)$, there exists a PPT algorithm $\mathcal{E}$ (called the* extractor*) which, given rewindable black-box access to $\tilde{\mathcal{P}}$ outputs a witness w such that $(x,w)\in\mathcal{R}$ in time $\mathsf{poly}(\lambda,(\tilde{\epsilon}-\epsilon)^{-1})$ with probability at least $1/2$.*
- (t,ζ)-zero-knowledge*, if for every PPT algorithm $\tilde{\mathcal{V}}$, there exists a PPT algorithm $\mathcal{S}$ (called the* simulator*) which, given the input statement $x\in\{0,1\}^\lambda$ and rewindable black-box access to $\tilde{\mathcal{V}}$, outputs a simulated transcript whose distribution is (t,ζ)-indistinguishable from $\mathrm{View}(\mathcal{P}(x,w),\tilde{\mathcal{V}}(x))$.*

2.4 Subset Sum Problems

We define hereafter two variants of the subset sum problem on which the security of our commitment scheme relies. The first one is the standard subset sum problem mentioned in the introduction, while the second one is a slightly stronger assumption that has already been used in cryptography (see, *e.g.* [5,27]).

Definition 4. *Let $t:\mathbb{N}\to\mathbb{N}$ and $\epsilon:\mathbb{N}\to[0,1]$. Let $\ell,m:\mathbb{N}\to\mathbb{N}$ and $\mathsf{modulus}$ be an algorithm which given $\lambda\in\mathbb{N}$ outputs an integer q of bit-length $m(\lambda)$. We consider the two following assumptions:*

- (t,ϵ)-(decision) subset-sum assumption for $(\ell,m,\mathsf{modulus})$*: for every algorithm $\mathcal{A}$, we have for all $\lambda\in\mathbb{N}$:*

$$\Pr\left[b=b'\;\middle|\;\begin{array}{l}q\xleftarrow{\$}\mathsf{modulus}(1^\lambda),\boldsymbol{\gamma}\xleftarrow{\$}[0,q-1]^{\ell(\lambda)},\boldsymbol{x}\xleftarrow{\$}\{0,1\}^{\ell(\lambda)},\\ t_0=\langle\boldsymbol{\gamma},\boldsymbol{x}\rangle \bmod q, t_1\xleftarrow{\$}[0,q-1], b\xleftarrow{\$}\{0,1\},\\ b'\xleftarrow{\$}\mathcal{A}(1^\lambda,q,\boldsymbol{\gamma},t_b)\end{array}\right]\leq\frac{1}{2}+\epsilon(\lambda)$$

  when $\mathcal{A}$ runs in time at most $t(\lambda)$ in this probabilistic computational game.
- (t,ϵ)-weighted knapsack assumption for $(\ell,m,\mathsf{modulus})$*: for every algorithm $\mathcal{A}$, we have for all $\lambda\in\mathbb{N}$:*

$$\Pr\left[\begin{array}{c}\langle\boldsymbol{\gamma},\boldsymbol{y}\rangle=0 \bmod q\\ \boldsymbol{y}\neq\mathbf{0}\in\{-1,0,1\}^{\ell(\lambda)}\end{array}\;\middle|\;\begin{array}{l}q\xleftarrow{\$}\mathsf{modulus}(1^\lambda),\boldsymbol{\gamma}\xleftarrow{\$}[0,q-1]^{\ell(\lambda)},\\ \boldsymbol{y}\xleftarrow{\$}\mathcal{A}(1^\lambda,q,\boldsymbol{\gamma})\end{array}\right]\leq\epsilon(\lambda)$$

  when $\mathcal{A}$ runs in time at most $t(\lambda)$ in this probabilistic computational game.

The search version of the subset sum assumption is polynomial-time equivalent to the decision version stated above. The hardness of these problems depends greatly on the *density* defined as $d(\lambda) = \ell(\lambda)/m(\lambda)$. If the density is too small (e.g. $d(\lambda) < 1/\ell(\lambda)$) or too large (e.g. $d(\lambda) > \ell(\lambda)$) then both problems can be solved in polynomial time (see e.g. [12] and references therein). Coster, Joux, LaMacchia, Odlyzko, Schnorr, and Stern [12] proved that the subset sum problem can be solved in polynomial-time with a single call to an oracle that can find the shortest vector in a special lattice of dimension $\ell(\lambda)+1$ if $d(\lambda) < 0.9408$ and Li and Ma proved a similar result for the weighted knapsack problem if $d(\lambda) < 0.488$. It is worth mentioning that these results do not break the assumptions in polynomial time since the best algorithm for finding the shortest vector in these lattices has computational complexity $2^{\Theta(\ell(\lambda))}$ (and cryptographic protocols relying on these problems with much smaller densities have been proposed, e.g. [21]).

In our construction, we will consider instances of these problems with density $d(\lambda) \simeq 1$ (i.e. $q \simeq 2^{\ell(\lambda)}$) for the subset sum problem since they are arguably the hardest ones [15]. This will result in instances for the weighted knapsack problem with density $d(\lambda) > 1$ and for conservative security, we will restrict ourselves to $d(\lambda) \leq 2$. In this case, lattice-based algorithms do not work and the best-known algorithms use very clever time-memory tradeoffs with the best algorithm due to Bonnetain, Bricout, Schrottenloher, and Shen [8] having time and memory complexities $\tilde{O}(2^{0.283\ell(\lambda)})$. These algorithms neglect the cost to access an exponential memory but even with this optimistic assumption, for $\ell(\lambda) = 256$, all known algorithms require at least a time complexity lower-bounded by 2^{128} operations or a memory of size at least 2^{64} bits. There also exists a vast literature on quantum algorithms for solving these problems (see [8] and references therein) and for $\ell(\lambda) = 256$, the best quantum algorithm requires about 2^{64} quantum operations and quantum memory.

3 String Commitments from Subset Sum Problems

3.1 Design Principle

In this section, we present our modified version of the bit-commitment based on the subset sum problem proposed in [15]. This new scheme enables commitments to bit-strings.

In [10], Brassard, Chaum, and Crépeau introduced the notion of *blob*, which is very similar to bit commitment, and presented an elegant construction based on the discrete-logarithm problem in groups of known prime order q (see also [9]). The commitment of a single bit consists of a group element (see Figure 1 (a) for an equivalent form of their commitment). Shortly afterward, Pedersen [24] introduced his celebrated commitment scheme that enables committing to an integer in $\mathbb{Z}_q$ with a single group element (see Figure 1 (c)). Impagliazzo and Naor [15] proposed a bit-commitment whose hiding and binding security properties rely on the subset sum problem. It has many similarities with the discrete-logarithm-based blob from [10,9] (see Figure 1 (b)).

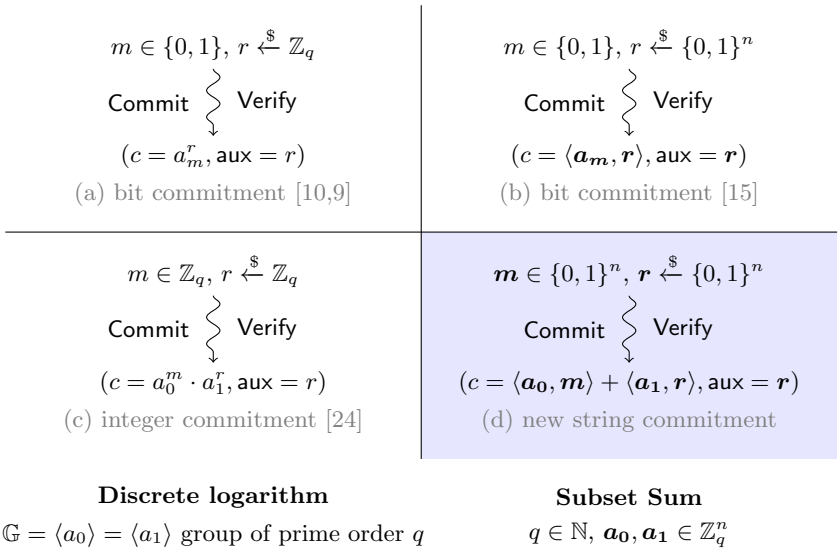


Fig. 1: Illustration of the Similarities between Commitment Schemes

To build our string commitment scheme, we push this analogy and propose a variant of Pedersen's protocol based on the subset sum (see Figure 1 (d)). The design principle is simple and maybe folklore but does not seem to have been published in this form (even if similar ideas have been used in lattice-based cryptography).

3.2 Formal Description and Security Analysis

Let $\ell, n, m : \mathbb{N} \to \mathbb{N}$ and let $\mathsf{modulus}$ be an algorithm which given $\lambda \in \mathbb{N}$ outputs an integer q of bit-length $m(\lambda)$. Typically, $\mathsf{modulus}$ outputs a random $m(\lambda)$-bit prime number or the unique integer $q = 2^{m(\lambda)-1}$. The function ℓ defines the message length while the function n defines the randomness length.

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. On input λ, the setup algorithm generates a modulus q by running $\mathsf{modulus}(1^\lambda)$ and picks uniformly at random $\boldsymbol{w} \in \mathbb{Z}_q^{\ell(\lambda)}$ and $\boldsymbol{s} \in \mathbb{Z}_q^{n(\lambda)}$. It outputs the public parameters $\mathsf{pp} = (q, \boldsymbol{w}, \boldsymbol{s})$ and the message space is $\mathcal{M} = \{0,1\}^{\ell(\lambda)}$.
- $\mathsf{Com}(\mathsf{pp}, m) \to (c, \mathsf{aux})$. On input pp and $\boldsymbol{m} \in \mathcal{M}$, the commit algorithm picks $\mathsf{aux} = \boldsymbol{r} \in \{0,1\}^{n(\lambda)}$ uniformly at random, computes $c = \langle \boldsymbol{w}, \boldsymbol{m} \rangle + \langle \boldsymbol{s}, \boldsymbol{r} \rangle \bmod q$ and outputs (c, aux).
- $\mathsf{Ver}(\mathsf{pp}, \boldsymbol{m}, c, \mathsf{aux}) \to b \in \{0,1\}$. On input pp, $\boldsymbol{m} \in \mathcal{M}$ and (c, aux), the verifier outputs 1 if $c = \langle \boldsymbol{w}, \boldsymbol{m} \rangle + \langle \boldsymbol{s}, \boldsymbol{r} \rangle \bmod q$ where $\boldsymbol{r} = \mathsf{aux} \in \{0,1\}^{n(\lambda)}$, and 0 otherwise.

We prove that our commitment scheme is hiding and binding assuming the hardness of the subset sum and the weighted knapsack problems (respectively) for different lengths in the subset sum problems.

Theorem 1. *Let $\ell, n, m : \mathbb{N} \to \mathbb{N}$ and let* modulus *be an algorithm which given $\lambda \in \mathbb{N}$ outputs an integer of bit-length $m(\lambda)$. This commitment scheme above is:*

1. *(t, ϵ)-computationally hiding if the $(t + O(\ell(\lambda)m(\lambda)), \epsilon)$- subset-sum assumption holds for $(\ell, m,$ modulus$)$;*
2. *(t, ϵ)-computationally binding if the $(t + O(\ell(\lambda) + n(\lambda)), \epsilon)$-weighted knapsack assumption holds for $(\ell + n, m,$ modulus$)$.*

Proof. Both security reductions are simple.

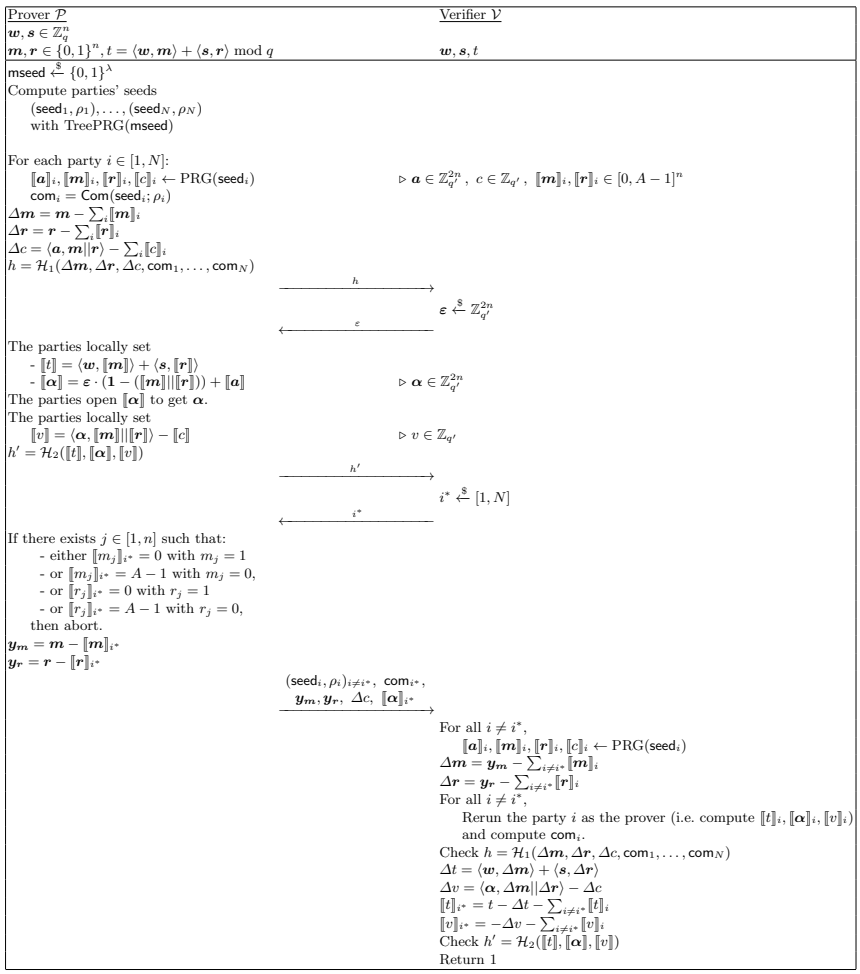
1. Let $\mathcal{A}$ be a (t, ϵ)-adversary against the hiding property of the commitment scheme. We construct a $(t + O(\ell(\lambda)m(\lambda)), \epsilon)$ adversary $\mathcal{B}$ breaking the decision subset sum assumption as follows. The algorithm $\mathcal{B}$ is given as inputs $(q, \boldsymbol{\gamma}, t)$ where $\boldsymbol{\gamma} \in \mathbb{Z}_q^{n(\lambda)}$. The algorithm $\mathcal{B}$ picks uniformly at random $\boldsymbol{w} \in \mathbb{Z}_q^{\ell(\lambda)}$, sets $\boldsymbol{s} = \boldsymbol{\gamma}$, and runs $\mathcal{A}_1$ on input $\mathsf{pp} = (q, \boldsymbol{w}, \boldsymbol{s})$. When $\mathcal{A}_1$ outputs two messages $\boldsymbol{m_0}, \boldsymbol{m_1} \in \{0,1\}^{\ell(\lambda)}$ and some state information s, the algorithm $\mathcal{B}$ picks uniformly at random a bit $b \in \{0,1\}$ and runs $\mathcal{A}_2$ on $c = \langle \boldsymbol{w}, \boldsymbol{m_b} \rangle + t \bmod q$ and s. Eventually, when $\mathcal{A}_2$ outputs some bit b', $\mathcal{B}$ outputs 0 if $b' = b$ and 1 otherwise. A routine argument shows that the advantage of $\mathcal{B}$ for the decision subset sum problem is identical to the one of $\mathcal{A}$ for breaking the hiding property.
2. Let $\mathcal{A}$ be a (t, ϵ)-adversary against the binding property of the commitment scheme. We construct a $(t + O(\ell(\lambda) + n(\lambda)), \epsilon)$ adversary $\mathcal{B}$ breaking the weighted knapsack assumption as follows. The algorithm $\mathcal{B}$ is given as inputs $(q, \boldsymbol{\gamma})$ where $\boldsymbol{\gamma} \in \mathbb{Z}_q^{\ell(\lambda)+n(\lambda)}$. It sets $\boldsymbol{w} = (\gamma_1, \ldots, \gamma_{\ell(\lambda)}) \in \mathbb{Z}_q^{\ell(\lambda)}$ and $\boldsymbol{s} = (\gamma_{\ell(\lambda)+1}, \ldots, \gamma_{\ell(\lambda)+n(\lambda)}) \in \mathbb{Z}_q^{n(\lambda)}$ and runs $\mathcal{A}$ on input $\mathsf{pp} = (q, \boldsymbol{w}, \boldsymbol{s})$. When $\mathcal{A}$ outputs $(\boldsymbol{m_1}, \boldsymbol{m_2}, \mathsf{aux}_1, \mathsf{aux}_2, c)$, we have $\boldsymbol{m_1} \neq \boldsymbol{m_2}$ and $\mathsf{Ver}(pp, m_1, c, aux_1) = \mathsf{Ver}(pp, m_2, c, aux_2) = 1$ with probability $\epsilon(\lambda)$. In this case, since $\boldsymbol{m_1} \neq \boldsymbol{m_2}$ and $(\boldsymbol{m_1}, \mathsf{aux}_1), (\boldsymbol{m_2}, \mathsf{aux}_2), \in \{0,1\}^{\ell(\lambda)+n(\lambda)}$, if $\mathcal{B}$ outputs the vector $\boldsymbol{y} = (\boldsymbol{m_1}, \mathsf{aux}_1) - (\boldsymbol{m_2}, \mathsf{aux}_2)$ (where the substraction is done coordinate-wise), it belongs to $\{-1, 0, 1\}^{\ell(\lambda)+n(\lambda)}$, is non-zero and satisfies $\langle \boldsymbol{\gamma}, \boldsymbol{y} \rangle = 0 \bmod q$ (and is thus a solution to the weighted knapsack problem $(q, \boldsymbol{\gamma})$).

□

The hiding property thus relies on the hardness of the subset sum problem with density $n(\lambda)/m(\lambda)$ while its binding property on the hardness of the weighted knapsack problem with density $(\ell(\lambda) + n(\lambda))/m(\lambda)$. In the following, to simplify the protocols, we consider the case where $n(\lambda) = m(\lambda)$ (*i.e.* density 1 subset sum) and $\ell(\lambda) = n(\lambda)$ (*i.e.* density 2 weighted knapsack). To lighten the notations, we henceforth denote $n = n(\lambda) = \ell(\lambda)$.

3.3 Zero-Knowledge Arguments for our Commitment

In this section, we present a zero-knowledge argument of knowledge for our string commitment. We apply readily the protocol recently proposed by Feneuil *et al.* [13] for the subset sum problem. It is based on the MPC-in-the-Head paradigm and is described in Protocol 1. We provide an explicit description of the protocol as we use it in the following sections but refer to [13] for details and precise security analysis.



Protocol 1: Zero-knowledge argument for string-commitment using batch product verification to prove binarity.

Context. We consider the binary relation $\mathcal{R} = \{(q, \boldsymbol{w}, \boldsymbol{s}, t), (\boldsymbol{m}, \boldsymbol{r})) \,|\, \langle \boldsymbol{w}, \boldsymbol{m} \rangle + \langle \boldsymbol{s}, \boldsymbol{r} \rangle = t \bmod q\}$ where $q \in \mathbb{N}$, $\boldsymbol{w}, \boldsymbol{s} \in \mathbb{Z}_q^n$, $t \in \mathbb{Z}_q$, and $\boldsymbol{m}, \boldsymbol{r} \in \{0,1\}^n$. Both the prover $\mathcal{P}$ and the verifier $\mathcal{V}$ know $(q, \boldsymbol{w}, \boldsymbol{s}, t)$ and $\mathcal{P}$ knows $(\boldsymbol{m}, \boldsymbol{r})$ and wants to convince the verifier of this fact. The protocol makes use of a PRG, a tree PRG [19], a commitment scheme $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Ver})$ (the one proposed in the previous section or any other efficient scheme) and two collision-resistant hash functions $\mathcal{H}_1$ and $\mathcal{H}_2$. The protocol involves two integer parameters A and a prime q' (that depends on n) that are known by $\mathcal{P}$ and $\mathcal{V}$.

MPC-in-the-Head. Feneuil *et al.*'s protocol [13] relies on the MPC-in-the-head paradigm [16]. The binary relation $\mathcal{R}$ defines an NP language and the membership of $(q, \boldsymbol{w}, \boldsymbol{s}, t)$ can be checked easily thanks to the knowledge of $(\boldsymbol{m}, \boldsymbol{r})$ by verifying the relations (1) $\langle \boldsymbol{w}, \boldsymbol{m} \rangle + \langle \boldsymbol{s}, \boldsymbol{r} \rangle = t \bmod q$ and (2) $\boldsymbol{m}, \boldsymbol{r} \in \{0,1\}^n$. To convince $\mathcal{V}$, $\mathcal{P}$ emulates "in their head" a $(N-1)$-private multi-party computation (MPC) protocol with N parties for the relations (1) and (2) where the witness $(\boldsymbol{m}, \boldsymbol{r})$ is shared among the N parties (for some parameter $N \in \mathbb{N}$).

To shorten the communication, shares and random coins used in the protocol are generated using the Tree PRG: $\mathcal{P}$ randomly and uniformly chooses a master seed mseed and constructs a tree of depth $\lceil \log N \rceil$ by expanding mseed into N subseeds as explained in Section 2. From these N subseeds $\mathsf{seed}_1, \ldots, \mathsf{seed}_N$, $\mathcal{P}$ constructs some additive integer secret sharing with shares in $[0, A-1]$ denoted as $[\![\cdot]\!]$ for the sharing itself and $[\![\cdot]\!]_i$ for the share of the i-th virtual player (*i.e.* a secret integer x is shared as $[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N) \in [0, A-1]^N$ such that $[\![x]\!]_1 + \cdots + [\![x]\!]_N = x$). Computation is done over $\mathbb{Z}_{q'}$ where q' is the smallest prime larger than A.

The verifier $\mathcal{V}$ challenges $\mathcal{P}$ to reveal the views of a random subset of $(N-1)$ parties by sending a challenge $i^* \in [1, N]$ to $\mathcal{P}$ who reveals all-but-one subseeds corresponding to parties $i \neq i^*$. In Feneuil *et al.*'s protocol [13], the integer secret sharing may reveal information and to avoid this $\mathcal{P}$ may abort but with probability at most $(1 - 1/A)^n$ (when sharing a n-coordinates vector). The size of A (and thus of q') has to be properly chosen to make this probability small in practice. Eventually, $\mathcal{V}$ recomputes the MPC protocol to check the views of the parties $i \neq i^*$ and the commitments. If all tests pass, $\mathcal{V}$ accepts the proof and the soundness error is close to $1/N$. To decrease it to a soundness error less than $2^{-\lambda}$, the protocol is simply repeated about $\tau \approx \lambda / \log(N)$ times.

The verification of (1) is linear modulo q' and is therefore free but proving (2) requires performing some multiplications in the MPC protocol (using the simple fact that $x \in \{0,1\}$ if and only if $x(1-x) = 0 \mod q'$). The verification of these multiplications can be realized following [4]. This implies a communication cost of $2\log(q')$ bits to prove one multiplication. Using a batched version of this verification protocol [18], one gets a communication cost of $(n+1)\log_2 q'$ for n multiplications. The soundness error of this protocol follows from the Schwartz-Zippel Lemma [28,25].

Security analysis. The following theorem from [13] states the completeness, soundness and zero-knowledge of Protocol 1.

Theorem 2 (Protocol 1). *Let the PRG used in Protocol 1 be (t, ε_{PRG})-secure and the commitment scheme* Com *be (t, ε_{Com})-hiding. Protocol 1 is a zero-knowledge proof of knowledge for the relation $\mathcal{R}$ with $(1-1/A)^{2n}$-completeness, $1/q' + 1/N - 1/Nq'$ soundness and $(t, \varepsilon_{PRG} + \varepsilon_{Com})$-zero-knowledge.*

Communication complexity. The communication cost (in bits) of the Protocol 1 with τ repetitions is

$$4\lambda + \tau \left[2n(\log_2(A-1) + \log_2(q')) + \log_2(q') + \lambda \log_2 N + 2\lambda\right].$$

Since the rejection rate after τ repetitions (i.e. that any of the τ repetition aborts) is given by $1-(1-1/A)^{2n\tau} \simeq 2n\tau/A$ where the approximation is tight when A is sufficiently large. Thus by taking $A = \Theta(n\tau)$, we get a (small) constant rejection probability.

Remark 1. Feneuil *et al.*'s [13] proposed a second approach to prove (2) using "*cut-and-choose*". It can be used to prove the knowledge of a commitment opening but does not adapt well for proving Boolean relations of committed values.

Remark 2. It is worth mentioning that our commitment and argument of knowledge of opening can be easily generalized to a proof of partial opening by revealing bits of the committed message, modifying the value of the commitment accordingly and proving the knowledge of the remaining hidden bits. This enables to provide a range proof of the committed message at no additional cost.

4 Zero-Knowledge Arguments for Boolean Relations

Using a batched version of the verification protocol [4,18] for multiplications of the form $xy = c$ with $x, y, c \in \mathbb{Z}_{q'}$ and c a public value, one gets a communication cost of $(n+1)\log_2 q'$ bits for n multiplications. In the following, we deal with multiplications of the form $xy = z$, where z is a linear combination of shared elements, and the communication cost remains $(n+1)\log_2 q'$ bits for n multiplications.

4.1 AND Gate

Coordinate-wise AND Gates. We first consider the case where three n-bits vectors $\boldsymbol{m^1}, \boldsymbol{m^2}, \boldsymbol{m^3}$ are committed and $\mathcal{P}$ wants to prove that $\boldsymbol{m^1} \cdot \boldsymbol{m^2} = \boldsymbol{m^3}$. Note that proving that $\boldsymbol{m^1}$ and $\boldsymbol{m^2}$ are binary and that $\boldsymbol{m^1} \cdot \boldsymbol{m^2} = \boldsymbol{m^3} \mod q'$ implies that $\boldsymbol{m^3}$ is binary and $\boldsymbol{m^1} \cdot \boldsymbol{m^2} = \boldsymbol{m^3}$. In addition, $\mathcal{P}$ has to prove that the three random vectors $\boldsymbol{r^1}, \boldsymbol{r^2}, \boldsymbol{r^3}$ used in the commitment are all binary (since no relation is proved between them). Using this approach, $\mathcal{P}$ has to prove $6n$ multiplications and therefore the argument requires sending $6n+1$ integers in $\mathbb{Z}_{q'}$ via [4,18].

Actually, it is possible to batch some verification equations and reduce this number from $6n+1$ to $5n+1$. Indeed, checking $\boldsymbol{m^1} \cdot \boldsymbol{m^2} = \boldsymbol{m^3} \mod q'$ and (for

instance) the binarity of $\boldsymbol{m^2}$ is equivalent (with a small soundness error coming from the Schwartz-Zippel Lemma) to check that

$$\lambda_1 \boldsymbol{m^2} \cdot (\boldsymbol{1} - \boldsymbol{m^2}) + \lambda_2 \boldsymbol{m^1} \cdot \boldsymbol{m^2} = \lambda_2 \boldsymbol{m^3} \mod q' \tag{1}$$

for $\lambda_1, \lambda_2 \in \mathbb{Z}_{q'}$ random elements chosen by $\mathcal{V}$. Hence, we can batch all the multiplications checking by verifying the component-wise product

$$(\boldsymbol{m^1}||\boldsymbol{r^1}||\boldsymbol{r^2}||\boldsymbol{r^3}||\boldsymbol{m^2}) \cdot ((\boldsymbol{1} - (\boldsymbol{m^1}||\boldsymbol{r^1}||\boldsymbol{r^2}||\boldsymbol{r^3}))||\lambda_1(\boldsymbol{1} - \boldsymbol{m^2}) + \lambda_2 \boldsymbol{m^1}) = (\boldsymbol{0}||\lambda_2 \boldsymbol{m^3})$$

and obtain Protocol 2.

Arbitrary AND Gates. Protocol 2 is similar to the protocols from [17,6,3,2] since it can be used only to prove multiplication coordinate-wise. We generalize it to obtain a more flexible protocol able to prove relations such as $m_i^1 \wedge m_j^2 = m_k^3$ for arbitrary coordinates $i, j, k \in [1, n]$.

Assume $\mathcal{P}$ has to prove the satisfiability of $K \geq 1$ AND gates with components belonging to $L \geq 1$ committed vectors $\{\boldsymbol{m^\ell}\}_{1 \leq \ell \leq L} \in \{0,1\}^n$. Suppose that there are $M \leq K$ AND gates such that each of them has at least one coordinate of a fixed committed vector $\boldsymbol{m^\ell}$ as input (for some $\ell \in [1, L]$). Assume these M gates are of the form $m_{x_k}^\ell \wedge m_{y_k}^{\ell_k} = m_{z_k}^{\ell'_k}$ for $k \in [1, M], \ell_k, \ell'_k \in [1, L], x_k, y_k, z_k \in [1, n]$ (again, we fix the vector $\boldsymbol{m^\ell}$). Moreover, as seen previously to check that $\boldsymbol{m^\ell}$ is binary, $\mathcal{V}$ can verify $\boldsymbol{m^\ell} \circ (\boldsymbol{1} - \boldsymbol{m^\ell}) = 0 \mod q'$. Then we can batch these verifications as

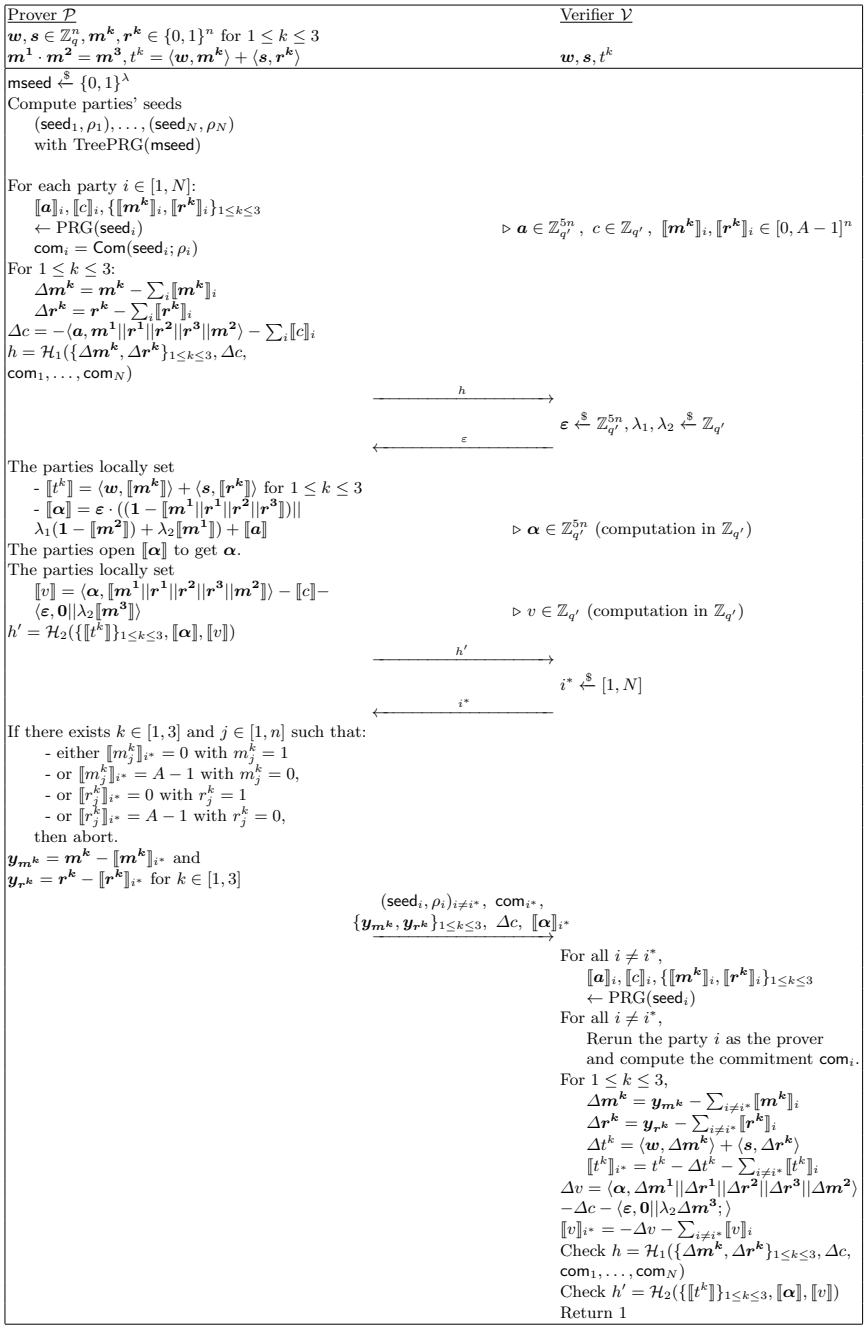
$$\lambda_0 \boldsymbol{m^\ell} \cdot (\boldsymbol{1} - \boldsymbol{m^\ell}) + \sum_{k=1}^{M} \lambda_k m_{x_k}^\ell m_{y_k}^{\ell_k} \boldsymbol{e_{x_k}} = \sum_{k=1}^{M} \lambda_k m_{z_k}^{\ell'_k} \boldsymbol{e_{x_k}} \mod q'$$

i.e.

$$\boldsymbol{m^\ell} \cdot [-\lambda_0 \boldsymbol{m^\ell} + \sum_{k=1}^{M} \lambda_k m_{y_k}^{\ell_k} \boldsymbol{e_{x_k}}] = -\lambda_0 \boldsymbol{m^\ell} + \sum_{k=1}^{M} \lambda_k m_{z_k}^{\ell'_k} \boldsymbol{e_{x_k}} \mod q' \tag{2}$$

where $\boldsymbol{e_i}$ is the i-th vector of the canonical basis of $\mathbb{Z}_{q'}^n$ and $\{\lambda_k\}_{0 \leq k \leq M} \in \mathbb{Z}_{q'}$ are random elements chosen by $\mathcal{V}$. Thus, $\mathcal{P}$ can batch all the gates' evaluation checking satisfying that at least one input for each of these gates belongs to the same committed vector. This batching can include the binary verification of this specific vector. In other words, the number of equations does not depend anymore on the number of gates (i.e. is independent of the distribution of AND gates over the committed bits). We obtain the generalized Protocol 3 as a direct extension of Protocol 2 (essentially the batching part is slightly different) which can be found in Appendix A.

Security analysis. The following theorems state the completeness, soundness, and zero-knowledge of Protocol 2 and Protocol 3. The proofs are similar to those in [13] and are omitted due to lack of space.



Protocol 2: Zero-knowledge argument for AND.

Theorem 3 (Protocol 2). *Let the PRG used in Protocol 1 be (t, ε_{PRG})-secure and the commitment scheme* Com *be (t, ε_{Com})-hiding. Protocol 2 is a zero-knowledge proof of knowledge for the relation $\mathcal{R}$ with $(1 - 1/A)^{6n}$-completeness, $1/q' + 1/N - 1/Nq'$ soundness and $(t, \varepsilon_{PRG} + \varepsilon_{Com})$-zero-knowledge.*

Theorem 4 (Protocol 3). *Let the PRG used in Protocol 1 be (t, ε_{PRG})-secure and the commitment scheme* Com *be (t, ε_{Com})-hiding. Protocol 3 is a zero-knowledge proof of knowledge for the relation $\mathcal{R}$ with $(1 - 1/A)^{2Ln}$-completeness, $1/q' + 1/N - 1/Nq'$ soundness and $(t, \varepsilon_{PRG} + \varepsilon_{Com})$-zero-knowledge.*

Remark 3. Note that Protocol 2 and 3 have the same soundness as Protocol 1. This follows from the Schwartz-Zippel Lemma, since the underlying multinomial still has the same degree after batching, and so it does not impact the soundness error. The rejection rates are bigger than in Protocol 1 since respectively $6n$ and $2Ln$ sharing over the integers are emulated instead of $2n$ in Protocol 1. This requires a slight increase of A (as shown in Table 1).

The communication cost (in bits) of Protocol 2 with τ repetitions is:

$$4\lambda + \tau \left[n(6 \log_2(A-1) + 5 \log_2(q')) + \log_2(q') + \lambda \log_2 N + 2\lambda \right],$$

and for the generalized Protocol 3:

$$4\lambda + \tau \left[2Ln(\log_2(A-1) + \log_2(q')) + \log_2(q') + \lambda \log_2 N + 2\lambda \right].$$

We notice that it does not depend on the number K of AND gates to prove.

4.2 XOR Gate

Coordinate-wise XOR Gates. We first consider, as above, the case where three n-bits vectors $\boldsymbol{m^1}, \boldsymbol{m^2}, \boldsymbol{m^3}$ are committed and $\mathcal{P}$ wants to prove that $\boldsymbol{m^1} \oplus \boldsymbol{m^2} = \boldsymbol{m^3}$ (coordinate-wise).

Let f be the polynomial $f(x) = 2x - x^2$ defined over $\mathbb{Z}_{q'}$ with $q' \geq 3$ a prime number. One can easily check that if $\boldsymbol{m^1}$ and $\boldsymbol{m^2}$ are binary vectors, then $f(\boldsymbol{m^1} + \boldsymbol{m^2}) \bmod q' = \boldsymbol{m^1} \oplus \boldsymbol{m^2} \in \{0, 1\}$. Thus, proving that $f(\boldsymbol{m^1} + \boldsymbol{m^2}) = \boldsymbol{m^3} \bmod q'$ in conjunction with the argument of knowledge of opening of the corresponding commitments, implies $\boldsymbol{m^1} \oplus \boldsymbol{m^2} = \boldsymbol{m^3}$.

With the same techniques as Protocol 2 for the $\wedge$ gate, we obtain Protocol 5 for bit-wise $\oplus$ gates which can be found in Appendix A.

Arbitrary XOR Gates. Again, this protocol is not enough flexible and can not be used to prove relations such as $m_i^1 \oplus m_j^2 = m_k^3$ for arbitrary $i, j, k \in [1, n]$, but we outline how to generalize it.

Assume that $\mathcal{P}$ has to prove the satisfiability of K XOR gates with inputs/output belonging to L committed vectors $\{\boldsymbol{m^\ell}\}_{1 \leq \ell \leq L} \in \{0, 1\}^n$. As for the AND gates, suppose that there are $M \leq K$ gates of the form $m_{x_k}^{\ell} \oplus m_{y_k}^{\ell_k} = m_{z_k}^{\ell'_k}$

for $k \in [1, M]$, $\ell_k, \ell'_k \in [1, L], x_k, y_k, z_k \in [1, n]$. We assume the binarity of each committed vector is checked during the protocol, so that

$$f(m_{x_k}^{\ell} + m_{y_k}^{\ell_k}) = 2(m_{x_k}^{\ell} + m_{y_k}^{\ell_k}) - (m_{x_k}^{\ell} + m_{y_k}^{\ell_k})^2 = m_{x_k}^{\ell} \oplus m_{y_k}^{\ell_k} = m_{z_k}^{\ell'_k} \mod q'.$$

Moreover, as seen previously, to check that $\boldsymbol{m}^{\boldsymbol{\ell}}$ is binary, $\mathcal{V}$ can verify $\boldsymbol{m}^{\boldsymbol{\ell}} \cdot (\mathbf{1} - \boldsymbol{m}^{\boldsymbol{\ell}}) = 0 \mod q'$. We can batch these equations as

$$\lambda_0 \boldsymbol{m}^{\boldsymbol{\ell}} \cdot (\mathbf{1} - \boldsymbol{m}^{\boldsymbol{\ell}}) + \sum_{k=1}^{K} \lambda_k \left(2(m_{x_k}^{\ell} + m_{y_k}^{\ell_k}) - (m_{x_k}^{\ell} + m_{y_k}^{\ell_k})^2\right) \boldsymbol{e}_{\boldsymbol{x_k}} = \sum_{k=1}^{K} \lambda_k m_{z_k}^{\ell'_k} \boldsymbol{e}_{\boldsymbol{x_k}} \mod q'.$$

If the binarity of $\boldsymbol{m}^{\boldsymbol{\ell_k}}$ and $\boldsymbol{m}^{\boldsymbol{\ell}}$ is proven elsewhere, $\mathcal{V}$ is convinced that $m_{y_k}^{\ell_k} m_{y_k}^{\ell_k} = m_{y_k}^{\ell_k} \mod q'$ and $m_{x_k}^{\ell} m_{x_k}^{\ell} = m_{x_k}^{\ell} \mod q'$. Hence, the batching equation becomes

$$\boldsymbol{m}^{\boldsymbol{\ell}} \cdot [-\lambda_0 \boldsymbol{m}^{\boldsymbol{\ell}} - 2 \sum_{k=1}^{M} \lambda_k m_{y_k}^{\ell_k} \boldsymbol{e}_{\boldsymbol{x_k}}] = -\lambda_0 \boldsymbol{m}^{\boldsymbol{\ell}} + \sum_{k=1}^{M} \lambda_k \left(m_{z_k}^{\ell'_k} - m_{y_k}^{\ell_k} - m_{x_k}^{\ell}\right) \boldsymbol{e}_{\boldsymbol{x_k}} \mod q', \quad (3)$$

where $\boldsymbol{e_i}$ is the i-th vector of the canonical basis of $\mathbb{Z}_{q'}^n$ and $\{\lambda_k\}_{0 \leq k \leq M} \in \mathbb{Z}_{q'}$ are random elements chosen by $\mathcal{V}$.

Security analysis. The theorems stating the completeness, soundness and zero-knowledge of the protocol for the bit-wise XOR and its generalization are the same as Theorems 3 and 4 (respectively). This follows directly from Remark 3.

The communication complexity (in bits) of the protocol for arbitrary XOR gates with τ repetitions is:

$$4\lambda + \tau \left[2Ln(\log_2(A-1) + \log_2(q')) + \log_2(q') + \lambda \log_2 N + 2\lambda\right],$$

while the one for the bit-wise XOR is a subcase when $L = 3$. We notice that the size is the same as Protocol 3 and is independent of K.

4.3 Instantiation and Performances

We present sets of parameters for an instantiation of our commitment scheme with $m(\lambda) = \ell(\lambda) = n(\lambda) = 256$ (*i.e.* with security based on density 1 subset-sum and density 2 weighted knapsack). We present performances for the component-wise protocols for AND and XOR gates. To decrease the rejection rate, we use a strategy introduced in [13] that consists in allowing $\mathcal{P}$ to abort in $0 \leq \eta < \tau$ out of the τ iterations and $\mathcal{V}$ accepts the proof if the prover can answer to $\tau - \eta$ challenges among the τ iterations. This relaxed proof has a significantly lower rejection rate (at the cost of a small increase of the soundness error).

Protocol	Parameters				Proof size	Rej. rate	Soundness err.
	τ	η	N	A			
Protocol 1 (Opening)	21	3	256	2^{13}	35.4 KB	0.035	133 bits
Protocol 1 (Opening)	19	2	256	2^{13}	33.3 KB	0.104	128 bits
Protocol 2 (AND)	21	3	256	2^{15}	98.9 KB	0.014	133 bits
Protocol 2 (AND)	19	2	256	2^{15}	93.4 KB	0.054	128 bits
Protocol 5 (XOR)	21	3	256	2^{15}	107.4 KB	0.014	133 bits
Protocol 5 (XOR)	19	2	256	2^{15}	101.3 KB	0.054	128 bits

Table 1: Comparison of performances with $n = 256$ and $q \approx 2^{256}$.

5 Verification of Circuit Evaluation

Let C be a Boolean circuit with $|C|$ gates (AND or XOR) and T input bits. Let $\boldsymbol{m} \in \{0,1\}^T$ and $v_1, \ldots, v_{|C|} \in \{0,1\}$ be committed elements such that $\boldsymbol{m}$ is an input that satisfy C and the v's are the outputs of each gates of C when evaluated on $\boldsymbol{m}$, *i.e.* $C(m_1, \ldots, m_T) = v_{|C|} = 1$. The prover $\mathcal{P}$ wants to prove that $\boldsymbol{m}$ indeed satisfies C. For this purpose, we will use protocols that have been presented in the previous sections. For simplicity, we assume without loss of generality that $T \leq n$. Since n bits can be committed via the same commitment (n is the size of the subset-sum instance), we need $|C|/n + 1$ string commitments. We introduce the following notation to simplify the batching equation: for $k \in [0, |C|/n]$, $\boldsymbol{v^0} = (\boldsymbol{m}||v_1|| \ldots ||v_{n-T}), \ldots, \boldsymbol{v^{|C|/n}} = (v_{|C|-T+1}|| \ldots ||v_{|C|}||\boldsymbol{0})$. Following the batching from Equation (2) and Equation (3), we can set $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ as follows so that the circuit satisfiability verification consists in checking that $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{z}$:

$$\boldsymbol{y} = (\boldsymbol{v^0}|| \ldots ||\boldsymbol{v^{|C|/n}}||\boldsymbol{r^0}|| \ldots ||\boldsymbol{r^{|C|/n}}),$$

$$\begin{aligned}\boldsymbol{x} = \Bigg(& -\lambda_0 \boldsymbol{v^0} + \sum_{i=1}^{n} \sum_{j=0}^{|C|/n} \sum_{k=1}^{n} \lambda_{v_k^j} \left(\delta_{0,i,j,k} v_k^j - 2\zeta_{0,i,j,k} v_k^j \right) \boldsymbol{e_i} || \ldots \\ & || - \lambda_{|C|/n} \boldsymbol{v^{|C|/n}} + \sum_{i=1}^{n} \sum_{j=0}^{|C|/n} \sum_{k=1}^{n} \lambda_{v_k^j} \left(\delta_{|C|/n,i,j,k} v_k^j - 2\zeta_{|C|/n,i,j,k} v_k^j \right) \boldsymbol{e_i} \\ & \qquad ||\boldsymbol{1} - \boldsymbol{r^1}|| \ldots ||\boldsymbol{1} - \boldsymbol{r^{|C|/n}} \Bigg)\end{aligned}$$

where $\boldsymbol{r^0}, \ldots, \boldsymbol{r^{|C|/n}}$ is the randomness used in the commitment and the vector $\boldsymbol{z}$ can be computed as a linear combination of $\boldsymbol{v^0}, \ldots, \boldsymbol{v^{|C|/n}}$. As above, $\boldsymbol{e_i}$ is the i-th vector of the canonical basis of $\mathbb{Z}_{q'}^n$, λ's are random public values chosen by the verifier $\mathcal{V}$, and the binary elements ζ and δ depend on the circuit structure, i.e. $\delta_{\ell,i,j,k} = 1$ if and only if $v_i^\ell \wedge v_k^j = v_p^u$ for some $u \in [0, |C|/n]$ and $v \in [1, n]$ (and $\zeta_{\ell,i,j,k} = 1$ if and only if $v_i^\ell \oplus v_k^j = v_p^u$). Hence, $\mathcal{V}$ has to check $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{z}$ to be convinced of the binarity of the vectors, and of the satisfiability of the circuit. The full protocol is given as Protocol 6 in Appendix A.

Theorem 5 (Protocol 6). *Let the PRG used in Protocol 6 be (t, ε_{PRG})-secure and the commitment scheme* Com *be (t, ε_{Com})-hiding. The protocol 6 is a zero-knowledge proof of knowledge for the relation $\mathcal{R}$ with $(1 - 1/A)^{2(|C|+n)}$-completeness, $1/q' + 1/N - 1/Nq'$ soundness and $(t, \varepsilon_{PRG} + \varepsilon_{Com})$-zero-knowledge.*

The communication cost (in bits) of Protocol 6 with τ repetitions is:

$$4\lambda + \tau \left[2(|C| + n)(\log(A - 1) + \log(q')) + \log(q') + \lambda \log N + 2\lambda\right].$$

With $n = 2\lambda$ and $A = \Theta((|C| + n)\tau)$ (for a small constant rejection probability), its asymptotic complexity is $\Theta\left(\frac{\lambda(|C|+\lambda)}{\log N} \log\left(\frac{\lambda(|C|+\lambda)}{\log N}\right) + \lambda^2\right)$. With $N = \Theta(\lambda)$ to minimize, we get asymptotic complexity $\tilde{\Theta}(\lambda|C| + \lambda^2)$ to be compared with $\tilde{\Theta}(|C|\lambda^2)$ in [17] (which can only prove Boolean relations bit-wise on binary strings and may result in a large overhead depending on the circuit considered).

Acknowledgements. The authors are supported in part by the French ANR SANGRIA project (ANR-21-CE39-0006).

References

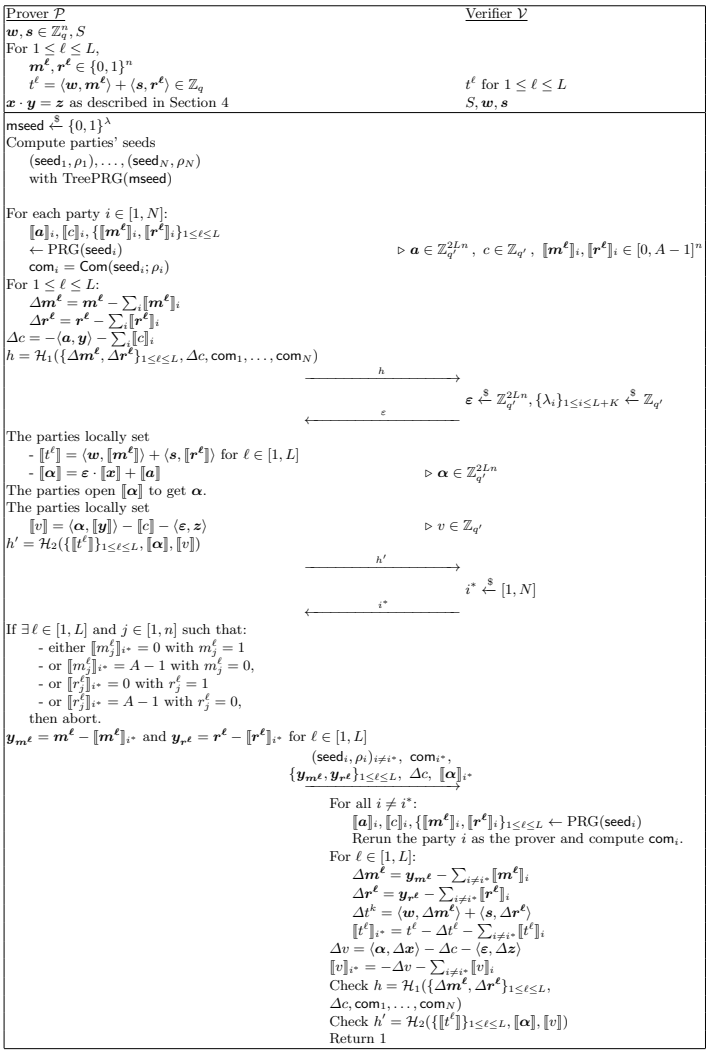
1. M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *29th ACM STOC*, pages 284–293, El Paso, TX, USA, May 4–6, 1997. ACM Press.
2. T. Attema, V. Lyubashevsky, and G. Seiler. Practical product proofs for lattice commitments. In D. Micciancio and T. Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 470–499, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Heidelberg, Germany.
3. C. Baum, I. Damgård, V. Lyubashevsky, S. Oechsner, and C. Peikert. More efficient commitments from structured lattice assumptions. In D. Catalano and R. De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 368–385, Amalfi, Italy, September 5–7, 2018. Springer, Heidelberg, Germany.
4. C. Baum and A. Nof. Concretely-efficient zero-knowledge arguments for arithmetic circuits and their application to lattice-based cryptography. In A. Kiayias, M. Kohlweiss, P. Wallden, and V. Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 495–526, Edinburgh, UK, May 4–7, 2020. Springer, Heidelberg, Germany.
5. M. Bellare and D. Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 163–192, Konstanz, Germany, May 11–15, 1997. Springer, Heidelberg, Germany.
6. F. Benhamouda, S. Krenn, V. Lyubashevsky, and K. Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In G. Pernul, P. Y. A. Ryan, and E. R. Weippl, editors, *ESORICS 2015, Part I*, volume 9326 of *LNCS*, pages 305–325, Vienna, Austria, September 21–25, 2015. Springer, Heidelberg, Germany.

7. M. Blum. Coin flipping by telephone - A protocol for solving impossible problems. In *COMPCON'82, Digest of Papers, Twenty-Fourth IEEE Computer Society International Conference, San Francisco, California, USA, February 22-25, 1982*, pages 133–137. IEEE Computer Society, 1982.
8. X. Bonnetain, R. Bricout, A. Schrottenloher, and Y. Shen. Improved classical and quantum algorithms for subset-sum. In S. Moriai and H. Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 633–666, Daejeon, South Korea, December 7–11, 2020. Springer, Heidelberg, Germany.
9. J. Boyar, S. A. Kurtz, and M. W. Krentel. A discrete logarithm implementation of perfect zero-knowledge blobs. *Journal of Cryptology*, 2(2):63–76, January 1990.
10. G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.
11. R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503, Montréal, Québec, Canada, May 19–21, 2002. ACM Press.
12. M. J. Coster, A. Joux, B. A. LaMacchia, A. M. Odlyzko, C. Schnorr, and J. Stern. Improved low-density subset sum algorithms. *Comput. Complex.*, 2:111–128, 1992.
13. T. Feneuil, J. Maire, M. Rivain, and D. Vergnaud. Zero-knowledge protocols for the subset sum problem from MPC-in-the-head with rejection. Cryptology ePrint Archive, Report 2022/223, 2022. `https://eprint.iacr.org/2022/223`.
14. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
15. R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, September 1996.
16. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM J. Comput.*, 39(3):1121–1152, 2009.
17. A. Jain, S. Krenn, K. Pietrzak, and A. Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 663–680, Beijing, China, December 2–6, 2012. Springer, Heidelberg, Germany.
18. D. Kales and G. Zaverucha. Efficient lifting for shorter zero-knowledge proofs and post-quantum signatures. Cryptology ePrint Archive, Paper 2022/588, 2022. `https://eprint.iacr.org/2022/588`.
19. J. Katz, V. Kolesnikov, and X. Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 525–537, Toronto, ON, Canada, October 15–19, 2018. ACM Press.
20. J. Kilian. *Uses of randomness in algorithms and protocols*. PhD thesis, Massachusetts Institute of Technology, 1989.
21. V. Lyubashevsky, A. Palacio, and G. Segev. Public-key cryptographic primitives provably as secure as subset sum. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 382–400, Zurich, Switzerland, February 9–11, 2010. Springer, Heidelberg, Germany.
22. R. C. Merkle and M. E. Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE Trans. Inf. Theory*, 24(5):525–530, 1978.
23. A. M. Odlyzko. The rise and fall of knapsack cryptosystems. Cryptology and computational number theory, Lect. Notes AMS Short Course, Boulder/CO (USA) 1989, Proc. Symp. Appl. Math. 42, 75-88 (1990)., 1990.
24. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Heidelberg, Germany.

25. J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.
26. P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134, Santa Fe, NM, USA, November 20–22, 1994. IEEE Computer Society Press.
27. R. Steinfeld, J. Pieprzyk, and H. Wang. On the provable security of an efficient RSA-based pseudorandom generator. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 194–209, Shanghai, China, December 3–7, 2006. Springer, Heidelberg, Germany.
28. R. Zippel. Probabilistic algorithms for sparse polynomials. In E. W. Ng, editor, *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposiumon Symbolic and Algebraic Computation, Marseille, France, June 1979, Proceedings*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.

A Description of Protocols 3, 5, and 6

In order to describe the circuit during Protocol 3, we set $S \leftarrow \emptyset$. Then construct S as follows: if $m_{x_k}^{\ell} \wedge m_{y_k}^{\ell_k} = m_{z_k}^{\ell'_k}$ for $k \in [1, M], \{\ell, \ell_k, \ell'_k\} \in [1, L]^3, \{x_k, y_k, z_k\} \in [1, n]^3$, then $S = S \cup \{(\ell, x_k; \ell_k, y_k; \ell'_k, z_k)\}$.



Protocol 3: Zero-knowledge argument for arbitrary AND gates.

Prover $\mathcal{P}$		Verifier $\mathcal{V}$
$\boldsymbol{w}, \boldsymbol{s} \in \mathbb{Z}_q^n, \boldsymbol{m^k}, \boldsymbol{r^k} \in \{0,1\}^n$ for $1 \le k \le 3$		
$\boldsymbol{m^1} \oplus \boldsymbol{m^2} = \boldsymbol{m^3}, t^k = \langle \boldsymbol{w}, \boldsymbol{m^k} \rangle + \langle \boldsymbol{s}, \boldsymbol{r^k} \rangle$		$\boldsymbol{w}, \boldsymbol{s}, t^k$ for $1 \le k \le 3$
$\mathsf{mseed} \xleftarrow{\$} \{0,1\}^\lambda$ Compute parties' seeds $(\mathsf{seed}_1, \rho_1), \ldots, (\mathsf{seed}_N, \rho_N)$ with TreePRG(mseed)		
For each party $i \in [1, N]$: $[\![\boldsymbol{a}]\!]_i, [\![c]\!]_i, \{[\![\boldsymbol{m^k}]\!]_i, [\![\boldsymbol{r^k}]\!]_i\}_{1 \le k \le 3}$ $\leftarrow \mathrm{PRG}(\mathsf{seed}_i)$ $\mathsf{com}_i = \mathsf{Com}(\mathsf{seed}_i; \rho_i)$		$\triangleright\ \boldsymbol{a} \in \mathbb{Z}_{q'}^{6n},\ c \in \mathbb{Z}_{q'},\ [\![\boldsymbol{m^k}]\!]_i, [\![\boldsymbol{r^k}]\!]_i \in [0, A-1]^n$
For $1 \le k \le 3$: $\Delta \boldsymbol{m^k} = \boldsymbol{m^k} - \sum_i [\![\boldsymbol{m^k}]\!]_i$ $\Delta \boldsymbol{r^k} = \boldsymbol{r^k} - \sum_i [\![\boldsymbol{r^k}]\!]_i$ $\Delta c = -\langle \boldsymbol{a}, \boldsymbol{m^1} \| \boldsymbol{m^2} \| \boldsymbol{r^1} \| \boldsymbol{r^2} \| \boldsymbol{r^3} \| \boldsymbol{m^1} + \boldsymbol{m^2} \rangle - \sum_i [\![c]\!]_i$ $h = \mathcal{H}_1(\{\Delta \boldsymbol{m^k}, \Delta \boldsymbol{r^k}\}_{1 \le k \le 3}, \Delta c, \mathsf{com}_1, \ldots, \mathsf{com}_N)$	$\xrightarrow{h}$	
		$\boldsymbol{\varepsilon} \xleftarrow{\$} \mathbb{Z}_{q'}^{6n}, \lambda_1, \lambda_2 \xleftarrow{\$} \mathbb{Z}_{q'}$
	$\xleftarrow{\varepsilon}$	
The parties locally set - $[\![t^k]\!] = \langle \boldsymbol{w}, [\![\boldsymbol{m^k}]\!] \rangle + \langle \boldsymbol{s}, [\![\boldsymbol{r^k}]\!] \rangle, 1 \le k \le 3$ - $[\![\boldsymbol{\alpha}]\!] = \boldsymbol{\varepsilon} \cdot ((\mathbf{1} - [\![\boldsymbol{m^1} \| \boldsymbol{m^2} \| \boldsymbol{r^1} \| \boldsymbol{r^2} \| \boldsymbol{r^3}]\!]) \| \| \boldsymbol{m^1} + \boldsymbol{m^2}) + [\![\boldsymbol{a}]\!]$		$\triangleright\ \boldsymbol{\alpha} \in \mathbb{Z}_{q'}^{6n}$ (computation in $\mathbb{Z}_{q'}$)
The parties open $[\![\boldsymbol{\alpha}]\!]$ to get $\boldsymbol{\alpha}$.		
The parties locally set $[\![v]\!] = \langle \boldsymbol{\alpha}, [\![\boldsymbol{m^1} \| \boldsymbol{m^2} \| \boldsymbol{r^1} \| \boldsymbol{r^2} \| \boldsymbol{r^3} \| \boldsymbol{m^1} + \boldsymbol{m^2}]\!] \rangle - [\![c]\!] - \langle \boldsymbol{\varepsilon}, \mathbf{0} \| [\![2(\boldsymbol{m^1} + \boldsymbol{m^2}) - \boldsymbol{m^3}]\!] \rangle$		$\triangleright\ v \in \mathbb{Z}_{q'}$ (computation in $\mathbb{Z}_{q'}$)
$h' = \mathcal{H}_2(\{[\![t^k]\!]\}_{1 \le k \le 3}, [\![\boldsymbol{\alpha}]\!], [\![v]\!])$	$\xrightarrow{h'}$	
		$i^* \xleftarrow{\$} [1, N]$
	$\xleftarrow{i^*}$	
If there exists $k \in [1,3]$ and $j \in [1,n]$ such that: - either $[\![m_j^k]\!]_{i^*} = 0$ with $m_j^k = 1$ - or $[\![m_j^k]\!]_{i^*} = A - 1$ with $m_j^k = 0$, - or $[\![r_j^k]\!]_{i^*} = 0$ with $r_j^k = 1$ - or $[\![r_j^k]\!]_{i^*} = A - 1$ with $r_j^k = 0$, then abort. $\boldsymbol{y_{m^k}} = \boldsymbol{m^k} - [\![\boldsymbol{m^k}]\!]_{i^*}$ and $\boldsymbol{y_{r^k}} = \boldsymbol{r^k} - [\![\boldsymbol{r^k}]\!]_{i^*}$ for $k \in [1,3]$		
	$(\mathsf{seed}_i, \rho_i)_{i \ne i^*},\ \mathsf{com}_{i^*},$ $\{\boldsymbol{y_{m^k}}, \boldsymbol{y_{r^k}}\}_{1 \le k \le 3},\ \Delta c,\ [\![\boldsymbol{\alpha}]\!]_{i^*}$ $\xrightarrow{\qquad\qquad\qquad\qquad\qquad\qquad}$	
		For all $i \ne i^*$, $[\![\boldsymbol{a}]\!]_i, [\![c]\!]_i, \{[\![\boldsymbol{m^k}]\!]_i, [\![\boldsymbol{r^k}]\!]_i\}_{1 \le k \le 3}$ $\leftarrow \mathrm{PRG}(\mathsf{seed}_i)$
		For all $i \ne i^*$, Rerun the party i as the prover and compute the commitment com_i.
		For $1 \le k \le 3$, $\Delta \boldsymbol{m^k} = \boldsymbol{y_{m^k}} - \sum_{i \ne i^*} [\![\boldsymbol{m^k}]\!]_i$ $\Delta \boldsymbol{r^k} = \boldsymbol{y_{r^k}} - \sum_{i \ne i^*} [\![\boldsymbol{r^k}]\!]_i$ $\Delta t^k = \langle \boldsymbol{w}, \Delta \boldsymbol{m^k} \rangle + \langle \boldsymbol{s}, \Delta \boldsymbol{r^k} \rangle$ $[\![t^k]\!]_{i^*} = t^k - \Delta t^k - \sum_{i \ne i^*} [\![t^k]\!]_i$
		$\Delta v = \langle \boldsymbol{\alpha}, \Delta \boldsymbol{m^1} \| \Delta \boldsymbol{m^2} \| \Delta \boldsymbol{r^1} \| \Delta \boldsymbol{r^2} \| \Delta \boldsymbol{r^3} \| \| \Delta \boldsymbol{m^1} + \Delta \boldsymbol{m^2} \rangle - \Delta c - \langle \boldsymbol{\varepsilon}, \mathbf{0} \| 2(\Delta \boldsymbol{m^1} + \Delta \boldsymbol{m^2}) - \Delta \boldsymbol{m^3}; \rangle$ $[\![v]\!]_{i^*} = -\Delta v - \sum_{i \ne i^*} [\![v]\!]_i$ Check $h = \mathcal{H}_1(\{\Delta \boldsymbol{m^k}, \Delta \boldsymbol{r^k}\}_{1 \le k \le 3}, \Delta c, \mathsf{com}_1, \ldots, \mathsf{com}_N)$ Check $h' = \mathcal{H}_2(\{[\![t^k]\!]\}_{1 \le k \le 3}, [\![\boldsymbol{\alpha}]\!], [\![v]\!])$ Return 1

Protocol 4: Zero-knowledge argument for XOR.

Prover $\mathcal{P}$		Verifier $\mathcal{V}$
$\boldsymbol{w}, \boldsymbol{s} \in \mathbb{Z}_q^n, \boldsymbol{m^k}, \boldsymbol{r^k} \in \{0,1\}^n$ for $1 \le k \le 3$		
$\boldsymbol{m^1} \oplus \boldsymbol{m^2} = \boldsymbol{m^3}, t^k = \langle \boldsymbol{w}, \boldsymbol{m^k} \rangle + \langle \boldsymbol{s}, \boldsymbol{r^k} \rangle$		$\boldsymbol{w}, \boldsymbol{s}, t^k$ for $1 \le k \le 3$
$\mathsf{mseed} \xleftarrow{\$} \{0,1\}^\lambda$ Compute parties' seeds $(\mathsf{seed}_1, \rho_1), \ldots, (\mathsf{seed}_N, \rho_N)$ with TreePRG(mseed)		
For each party $i \in [1, N]$: $[\![\boldsymbol{a}]\!]_i, [\![c]\!]_i, \{[\![\boldsymbol{m^k}]\!]_i, [\![\boldsymbol{r^k}]\!]_i\}_{1 \le k \le 3}$ $\leftarrow \mathrm{PRG}(\mathsf{seed}_i)$ $\mathsf{com}_i = \mathsf{Com}(\mathsf{seed}_i; \rho_i)$		$\triangleright\ \boldsymbol{a} \in \mathbb{Z}_{q'}^{6n},\ c \in \mathbb{Z}_{q'},\ [\![\boldsymbol{m^k}]\!]_i, [\![\boldsymbol{r^k}]\!]_i \in [0, A-1]^n$
For $1 \le k \le 3$: $\Delta \boldsymbol{m^k} = \boldsymbol{m^k} - \sum_i [\![\boldsymbol{m^k}]\!]_i$ $\Delta \boldsymbol{r^k} = \boldsymbol{r^k} - \sum_i [\![\boldsymbol{r^k}]\!]_i$ $\Delta c = -\langle \boldsymbol{a}, \boldsymbol{m^1} \| \boldsymbol{m^2} \| \boldsymbol{r^1} \| \boldsymbol{r^2} \| \boldsymbol{r^3} \| \boldsymbol{m^1} + \boldsymbol{m^2} \rangle - \sum_i [\![c]\!]_i$ $h = \mathcal{H}_1(\{\Delta \boldsymbol{m^k}, \Delta \boldsymbol{r^k}\}_{1 \le k \le 3}, \Delta c, \mathsf{com}_1, \ldots, \mathsf{com}_N)$	$\xrightarrow{h}$	
		$\boldsymbol{\varepsilon} \xleftarrow{\$} \mathbb{Z}_{q'}^{6n}, \lambda_1, \lambda_2 \xleftarrow{\$} \mathbb{Z}_{q'}$
	$\xleftarrow{\varepsilon}$	
The parties locally set - $[\![t^k]\!] = \langle \boldsymbol{w}, [\![\boldsymbol{m^k}]\!] \rangle + \langle \boldsymbol{s}, [\![\boldsymbol{r^k}]\!] \rangle, 1 \le k \le 3$ - $[\![\boldsymbol{\alpha}]\!] = \boldsymbol{\varepsilon} \cdot ((\mathbf{1} - [\![\boldsymbol{m^1} \| \boldsymbol{m^2} \| \boldsymbol{r^1} \| \boldsymbol{r^2} \| \boldsymbol{r^3}]\!]) \| \| \boldsymbol{m^1} + \boldsymbol{m^2}) + [\![\boldsymbol{a}]\!]$		$\triangleright\ \boldsymbol{\alpha} \in \mathbb{Z}_{q'}^{6n}$ (computation in $\mathbb{Z}_{q'}$)
The parties open $[\![\boldsymbol{\alpha}]\!]$ to get $\boldsymbol{\alpha}$.		
The parties locally set $[\![v]\!] = \langle \boldsymbol{\alpha}, [\![\boldsymbol{m^1} \| \boldsymbol{m^2} \| \boldsymbol{r^1} \| \boldsymbol{r^2} \| \boldsymbol{r^3} \| \boldsymbol{m^1} + \boldsymbol{m^2}]\!] \rangle - [\![c]\!] - \langle \boldsymbol{\varepsilon}, \mathbf{0} \| [\![2(\boldsymbol{m^1} + \boldsymbol{m^2}) - \boldsymbol{m^3}]\!] \rangle$		$\triangleright\ v \in \mathbb{Z}_{q'}$ (computation in $\mathbb{Z}_{q'}$)
$h' = \mathcal{H}_2(\{[\![t^k]\!]\}_{1 \le k \le 3}, [\![\boldsymbol{\alpha}]\!], [\![v]\!])$	$\xrightarrow{h'}$	
		$i^* \xleftarrow{\$} [1, N]$
	$\xleftarrow{i^*}$	
If there exists $k \in [1,3]$ and $j \in [1,n]$ such that: - either $[\![m_j^k]\!]_{i^*} = 0$ with $m_j^k = 1$ - or $[\![m_j^k]\!]_{i^*} = A - 1$ with $m_j^k = 0$, - or $[\![r_j^k]\!]_{i^*} = 0$ with $r_j^k = 1$ - or $[\![r_j^k]\!]_{i^*} = A - 1$ with $r_j^k = 0$, then abort. $\boldsymbol{y_{m^k}} = \boldsymbol{m^k} - [\![\boldsymbol{m^k}]\!]_{i^*}$ and $\boldsymbol{y_{r^k}} = \boldsymbol{r^k} - [\![\boldsymbol{r^k}]\!]_{i^*}$ for $k \in [1,3]$		
	$(\mathsf{seed}_i, \rho_i)_{i \ne i^*},\ \mathsf{com}_{i^*},$ $\{\boldsymbol{y_{m^k}}, \boldsymbol{y_{r^k}}\}_{1 \le k \le 3},\ \Delta c,\ [\![\boldsymbol{\alpha}]\!]_{i^*}$ $\xrightarrow{\qquad\qquad\qquad\qquad\qquad\qquad}$	
		For all $i \ne i^*$, $[\![\boldsymbol{a}]\!]_i, [\![c]\!]_i, \{[\![\boldsymbol{m^k}]\!]_i, [\![\boldsymbol{r^k}]\!]_i\}_{1 \le k \le 3}$ $\leftarrow \mathrm{PRG}(\mathsf{seed}_i)$
		For all $i \ne i^*$, Rerun the party i as the prover and compute the commitment com_i.
		For $1 \le k \le 3$, $\Delta \boldsymbol{m^k} = \boldsymbol{y_{m^k}} - \sum_{i \ne i^*} [\![\boldsymbol{m^k}]\!]_i$ $\Delta \boldsymbol{r^k} = \boldsymbol{y_{r^k}} - \sum_{i \ne i^*} [\![\boldsymbol{r^k}]\!]_i$ $\Delta t^k = \langle \boldsymbol{w}, \Delta \boldsymbol{m^k} \rangle + \langle \boldsymbol{s}, \Delta \boldsymbol{r^k} \rangle$ $[\![t^k]\!]_{i^*} = t^k - \Delta t^k - \sum_{i \ne i^*} [\![t^k]\!]_i$
		$\Delta v = \langle \boldsymbol{\alpha}, \Delta \boldsymbol{m^1} \| \Delta \boldsymbol{m^2} \| \Delta \boldsymbol{r^1} \| \Delta \boldsymbol{r^2} \| \Delta \boldsymbol{r^3} \| \Delta \boldsymbol{m^1} + \Delta \boldsymbol{m^2} \rangle - \Delta c - \langle \boldsymbol{\varepsilon}, \mathbf{0} \| 2(\Delta \boldsymbol{m^1} + \Delta \boldsymbol{m^2}) - \Delta \boldsymbol{m^3}; \rangle$ $[\![v]\!]_{i^*} = -\Delta v - \sum_{i \ne i^*} [\![v]\!]_i$ Check $h = \mathcal{H}_1(\{\Delta \boldsymbol{m^k}, \Delta \boldsymbol{r^k}\}_{1 \le k \le 3}, \Delta c, \mathsf{com}_1, \ldots, \mathsf{com}_N)$ Check $h' = \mathcal{H}_2(\{[\![t^k]\!]\}_{1 \le k \le 3}, [\![\boldsymbol{\alpha}]\!], [\![v]\!])$ Return 1

Protocol 5: Zero-knowledge argument for XOR.

Prover $\mathcal{P}$		Verifier $\mathcal{V}$
C, $\boldsymbol{w}, \boldsymbol{s} \in \mathbb{Z}_{q'}^n$ For $0 \le \ell \le \lvert C\rvert/n$ $\quad \boldsymbol{v}^{\boldsymbol{\ell}}, \boldsymbol{r}^{\boldsymbol{\ell}} \in \{0,1\}^n$ $\quad t^\ell = \langle \boldsymbol{w}, \boldsymbol{v}^{\boldsymbol{\ell}} \rangle + \langle \boldsymbol{s}, \boldsymbol{r}^{\boldsymbol{\ell}} \rangle$ $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{z}$ as described in Section 5		$C, \boldsymbol{w}, \boldsymbol{s}, t^\ell$ for $0 \le \ell \le \lvert C\rvert/n$
$\mathsf{mseed} \xleftarrow{\$} \{0,1\}^\lambda$ Compute parties' seeds $\quad (\mathsf{seed}_1, \rho_1), \ldots, (\mathsf{seed}_N, \rho_N)$ $\quad$ with TreePRG(mseed) For each party $i \in [1, N]$: $\quad [\![\boldsymbol{a}]\!]_i, [\![c]\!]_i, \{[\![\boldsymbol{v}^{\boldsymbol{\ell}}]\!]_i, [\![\boldsymbol{r}^{\boldsymbol{\ell}}]\!]_i\}_{0 \le \ell \le \lvert C\rvert/n}$ $\quad \leftarrow \mathrm{PRG}(\mathsf{seed}_i)$ $\quad \mathsf{com}_i = \mathsf{Com}(\mathsf{seed}_i; \rho_i)$ For $0 \le \ell \le \lvert C\rvert/n$: $\quad \Delta\boldsymbol{r}^{\boldsymbol{\ell}} = \boldsymbol{r}^{\boldsymbol{\ell}} - \sum_i [\![\boldsymbol{r}^{\boldsymbol{\ell}}]\!]_i$ $\quad \Delta\boldsymbol{v}^{\boldsymbol{\ell}} = \boldsymbol{v}^{\boldsymbol{\ell}} - \sum_i [\![\boldsymbol{v}^{\boldsymbol{\ell}}]\!]_i$ $\Delta c = -\langle \boldsymbol{a}, \boldsymbol{y} \rangle - \sum_i [\![c]\!]_i$ $h = \mathcal{H}_1(\{\Delta\boldsymbol{r}^{\boldsymbol{\ell}}, \Delta\boldsymbol{v}^{\boldsymbol{\ell}}\}_{0 \le \ell \le \lvert C\rvert/n}, \Delta c, \mathsf{com}_1, \ldots, \mathsf{com}_N)$		
	$\xrightarrow{h}$	
		$\boldsymbol{\varepsilon} \xleftarrow{\$} \mathbb{Z}_{q'}^{2(\lvert C\rvert+n)}$ $\{\lambda_i\}_{0 \le i \le \lvert C\rvert(1+1/n)} \xleftarrow{\$} \mathbb{Z}_{q'}$
	$\xleftarrow{\boldsymbol{\varepsilon}}$	
The parties locally set $\quad$ - $[\![t^\ell]\!] = \langle \boldsymbol{w}, [\![v^\ell]\!] \rangle + \langle \boldsymbol{s}, [\![\boldsymbol{r}^{\boldsymbol{\ell}}]\!] \rangle$ for $\ell \in [0, \lvert C\rvert/n]$ $\quad$ - $[\![\boldsymbol{\alpha}]\!] = \boldsymbol{\varepsilon} \cdot [\![\boldsymbol{x}]\!] + [\![\boldsymbol{a}]\!]$	$\triangleright\ \boldsymbol{\alpha} \in \mathbb{Z}_{q'}^{2(\lvert C\rvert+n)}$	
The parties open $[\![\boldsymbol{\alpha}]\!]$ to get $\boldsymbol{\alpha}$.		
The parties locally set $[\![v]\!] = \langle \boldsymbol{\alpha}, [\![\boldsymbol{y}]\!] \rangle - [\![c]\!] - \langle \boldsymbol{\varepsilon}, [\![\boldsymbol{z}]\!] \rangle$	$\triangleright\ v \in \mathbb{Z}_{q'}$	
$h' = \mathcal{H}_2(\{[\![t^\ell]\!]\}_{0 \le \ell \le \lvert C\rvert/n}, [\![\boldsymbol{\alpha}]\!], [\![v]\!])$		
	$\xrightarrow{h'}$	
		$i^* \xleftarrow{\$} [1, N]$
	$\xleftarrow{i^*}$	
If $\exists\, \ell \in [0, \lvert C\rvert/n], j \in [1, n]$ such that: $\quad$ - either $[\![v_j^\ell]\!]_{i^*} = 0$ with $v_j^\ell = 1$ $\quad$ - or $[\![v_j^\ell]\!]_{i^*} = A - 1$ with $v_j^\ell = 0$, $\quad$ - or $[\![r_j^\ell]\!]_{i^*} = 0$ with $r_j^\ell = 1$ $\quad$ - or $[\![r_j^\ell]\!]_{i^*} = A - 1$ with $r_j^\ell = 0$, $\quad$ then abort.		
$\boldsymbol{y}_{\boldsymbol{v}^{\boldsymbol{\ell}}} = \boldsymbol{v}^{\boldsymbol{\ell}} - [\![\boldsymbol{v}^{\boldsymbol{\ell}}]\!]_{i^*}$ and $\boldsymbol{y}_{\boldsymbol{r}^{\boldsymbol{\ell}}} = \boldsymbol{r}^{\boldsymbol{\ell}} - [\![\boldsymbol{r}^{\boldsymbol{\ell}}]\!]_{i^*}$		
	$(\mathsf{seed}_i, \rho_i)_{i \ne i^*},\ \mathsf{com}_{i^*},$ $\{\boldsymbol{y}_{\boldsymbol{v}^{\boldsymbol{\ell}}}, \boldsymbol{y}_{\boldsymbol{r}^{\boldsymbol{\ell}}}\}_{0 \le \ell \le \lvert C\rvert/n},\ \Delta c,\ [\![\boldsymbol{\alpha}]\!]_{i^*}$ $\xrightarrow{\qquad\qquad\qquad\qquad\qquad\qquad}$	
	For all $i \ne i^*$, $\quad [\![\boldsymbol{a}]\!]_i, [\![c]\!]_i, \{[\![\boldsymbol{v}^{\boldsymbol{\ell}}]\!]_i, [\![\boldsymbol{r}^{\boldsymbol{\ell}}]\!]_i\}_{0 \le \ell \le \lvert C\rvert/n} \leftarrow \mathrm{PRG}(\mathsf{seed}_i)$ $\quad$ Rerun the party i as the prover and compute com_i. For $0 \le \ell \le \lvert C\rvert/n$, $\quad \Delta\boldsymbol{v}^{\boldsymbol{\ell}} = \boldsymbol{y}_{\boldsymbol{v}^{\boldsymbol{\ell}}} - \sum_{i \ne i^*} [\![\boldsymbol{v}^{\boldsymbol{\ell}}]\!]_i$, $\Delta\boldsymbol{r}^{\boldsymbol{\ell}} = \boldsymbol{y}_{\boldsymbol{r}^{\boldsymbol{\ell}}} - \sum_{i \ne i^*} [\![\boldsymbol{r}^{\boldsymbol{\ell}}]\!]_i$ $\quad \Delta t^\ell = \langle \boldsymbol{w}, \Delta\boldsymbol{v}^{\boldsymbol{\ell}} \rangle + \langle \boldsymbol{s}, \Delta\boldsymbol{r}^{\boldsymbol{\ell}} \rangle$ $\quad [\![t^\ell]\!]_{i^*} = t^\ell - \Delta t^\ell - \sum_{i \ne i^*} [\![t^\ell]\!]_i$ $\Delta v = \langle \boldsymbol{\alpha}, \Delta\boldsymbol{x} \rangle - \Delta c - \langle \boldsymbol{\varepsilon}, \Delta\boldsymbol{z} \rangle$, $[\![v]\!]_{i^*} = -\Delta v - \sum_{i \ne i^*} [\![v]\!]_i$ Check $h = \mathcal{H}_1(\{\Delta\boldsymbol{v}^{\boldsymbol{\ell}}, \Delta\boldsymbol{r}^{\boldsymbol{\ell}}\}_{0 \le \ell \le \lvert C\rvert/n}, \Delta c, \mathsf{com}_1, \ldots, \mathsf{com}_N)$ Check $h' = \mathcal{H}_2(\{[\![t^\ell]\!]\}_\ell, [\![v]\!])$ Return 1	

Protocol 6: Zero-knowledge argument for Circuit Satisfiability.