EPiC Series in Computing

Volume 106, 2025, Pages 1–15

Proceedings of the 20th Asia Joint Conference on Information Security

A Generic Zero-Knowledge Range Argument with Preprocessing

Yuki Sawai[1,2], Kyoichi Asano[2], Yohei Watanabe[2], and Mitsugu Iwamoto[2]

[1] DENSO CORPORATION, Aichi, Japan
yuki.sawai.j8j@jp.denso.com
[2] The University of Electro-Communications, Tokyo, Japan
{sawaiyuki, k.asano, watanabe, mitsugu}@uec.ac.jp

Abstract

Range arguments are a type of zero-knowledge proofs that aim to prove that a prover's committed value falls within a specified range for a verifier. Previously, most range arguments were constructed based on the DLOG assumption, and hence, exponentiation operation is required for proof generation and verification. In addition, it is generally known that splitting a zero-knowledge proof protocol into a preprocessing phase and an online phase makes computation after fixing the input efficient. Still, such protocol has yet to be known for range arguments. This paper proposes an efficient range arguments protocol with a preprocessing phase. Our proposal takes a new approach by using arithmetic circuits to express the constraints that the prover must prove. The prover (resp. verifier) can generate (resp. verify) a part of proof based on multiplication and addition operations instead of exponentiation operations. Our range argument is a generic construction that does not rely on any particular mathematical assumptions, which enables us to construct a post-quantum range argument. The implementation evaluation shows that the total computation time for the prover and verifier in the online phase is efficient compared to Bulletproofs, one of the state-of-the-art range proofs. Especially, the prover computation is efficient.

1 Introduction

Zero-knowledge proofs (ZKPs) allow a prover to convince a verifier of the truth of a statement, without revealing any further information. They are crucial building blocks for various secure applications such as confidential transactions [4], signature schemes, and anonymous credential systems, since the first proposal in 1985 [12].

Range arguments are a type of zero-knowledge proofs that aim to prove that a prover's committed value falls within a specified range for a verifier. Since the first practical range proof construction was proposed [3], range arguments have been extensively used in various applications, e.g., anonymous credentials (to prove that a secret credential attribute, i.e., user age, falls in a specific range) [7], confidential transactions [14] on blockchain systems, e-cash [6] and electronic voting systems [1]. Computationally efficient range arguments are one of the leading research topics of range arguments. Previously, most range arguments were constructed based

A. Yamada, H.K. Kim, Y. Wang and T.-T. Tsai (eds.), AsiaJCIS 2025 (EPiC Series in Computing, vol. 106), pp. 1–15

on the DLOG assumption and hence, exponentiation operation is required for proof generation and verification [5, 17, 10, 9, 8]. Since exponentiation is a heavy process compared to multiplication and addition operations, computation will become more efficient if proof generation (resp. verification) can be executed by multiplication and addition operations. In addition, it is generally known that splitting a zero-knowledge proof protocol into a preprocessing phase and an online phase makes computation after fixing the input efficient [18, 19]. Suppose the communication destination is fixed in advance, such as an electronic voting systems provider for electronic voting or a particular e-cash system provider. In that case, such offline-online protocol can be applied, but such protocol has yet to be known for range arguments.

1.1 Our Contribution

We propose a range argument protocol with a preprocessing phase. Compared to conventional protocols based on the DLOG assumption [5, 17] and integer commitment [10, 9, 8], our protocol is new in that it is based on arithmetic circuits. We can split our range argument protocol into a preprocessing phase and an online phase. This makes computation after fixing the input efficient. Our range argument is efficient because, in addition to splitting into the preprocessing phase, the prover (resp. verifier) generates (resp. verifies) a part of proof based on information-theoretic message authentication codes (IT-MACs), which can be computed by multiplication and addition operations. It makes proof generation (resp. verification) more efficient computationally than computed by exponential operations. Moreover, our range argument is a generic construction that does not rely on any particular mathematical assumptions. In more detail, our range argument relies on mathematical assumptions of a vector oblivious linear evaluation (VOLE) and commitment that we employ. We can employ various types of VOLE construction without any restrictions. However, it is not the same for commitments. That is, commitments need additional properties to construct our range arguments securely. We newly define these properties and prove that any homomorphic commitment satisfies these properties. Therefore, by adapting the standard Pedersen commitment [16], our range argument is no need to use a large class of groups as in square decomposition-based approaches [9, 8] to obtain 128-bit security. Moreover, we can construct a post-quantum range argument by adapting quantum-resistant VOLE like [18], which is based on LPN assumption, and that of commitment like [2], which is based on lattice assumptions. Moreover, our range argument does not require a trusted setup. We can make our range argument non-interactive using Fiat-Shamir heuristic [11].

The implementation evaluation shows that the total computation time for the prover and verifier in the online phase is efficient compared to Bulletproofs, one of the state-of-the-art methods. Especially, the prover computation is efficient. The verifier computation is not as fast as Bulletproofs, but it is still efficient enough.

2 Technical Overview

Our main idea is to express the constraints that the prover must prove in range arguments as arithmetic circuits and to adapt them to QuickSilver [19], which is one of ZKP for an arithmetic circuit (ZKP-AC). ZKP-AC allows a prover with witness w to prove $\mathcal{C}(w) = 0$ to a verifier for some public arithmetic circuit $\mathcal{C}$ without revealing any additional information beyond the fact that $\mathcal{C}(w) = 0$. QuickSilver generates (resp. verifies) the proof by using VOLE correlations. This type of ZKP-AC is called VOLE-based zero-knowledge proof (VOLE-based ZKP). QuickSilver has computationally efficient features, such as preprocessing and proof generation (resp. verification) by multiplication and addition operations. We can enjoy these

features if we can adapt constraints of range arguments to QuickSilver, but it is actually non-trivial for the following two reasons. First, unlike in ZKP-AC, we must assume that the prover's witness has already been committed before fixing the input in the protocol. In other words, the prover must prove not only that (1) the witness w lies within a certain range, but also that (2) w is indeed the committed value. This makes a QuickSilver-based range argument non-trivial; it is unclear how we prove (2) in the QuickSilver framework, i.e., with arithmetic circuits. Second, we must express the constraints that the prover proves in range arguments as arithmetic circuits to prove (1). Since arithmetic circuits consist solely of addition and multiplication operations, we have to represent the range-argument constraints as a function using only addition and multiplication.

Based on the above observation, we propose a new range argument based on QuickSilver effectively combined with commitment schemes. We newly define extended versions of hiding and binding properties to construct our range arguments securely. We prove that any homomorphic commitment satisfies these properties, which makes our range argument adaptable to any homomorphic commitment. Moreover, we introduce a new representation of arithmetic circuits for the constraints of the range argument. Since QuickSilver requires the same number of interactions as there are multiplication gates in the arithmetic circuits,[1] it is crucial to design an arithmetic circuit for range arguments with as few multiplication gates as possible. Our arithmetic circuit only requires n multiplication gates (that require interactions) for the range $[0, 2^n - 1]$, based on the prior works [5, 17]. We give technical overviews of the above technical contributions as follows.

Adapting Commitment Schemes. In QuickSilver, a prover masks his witness w by (VOLE) randomness μ and sends $\sigma := w - \mu$ to a verifier. Afterward, the prover generates MAC tags for w as a proof, and the verifier generates the corresponding MAC key using σ in a zero-knowledge manner. The prover homomorphically computes a MAC tag for $\mathcal{C}(w) = 0$ and sends it to the verifier. The verifier, in turn, homomorphically computes the corresponding key for $\mathcal{C}(w) = 0$. As a result, the MAC verification will accept the MAC tag if and only if both the prover and verifier have followed the protocol correctly. To construct a range argument based on QuickSilver, we need to force the prover to bind the message w in the commitment c_w to the witness w used for $\sigma := w - \mu$. Otherwise, even if the prover uses the witness w' in σ that is not equal to the committed w, the verifier cannot verify $w' \neq w$ and incorrectly accepts w'. This makes QuickSilver-based range arguments challenging since the language for the range argument is an NP language, given that w is committed and cannot be altered due to the commitment scheme.

Given the above discussion, we add a verification procedure that checks the prover's input w is equivalent to the commited value w with the aid of the homomorphic property of the commitment scheme. The prover sends a commitment $\mathsf{c}_{-\mu} := \mathsf{Com}(-\mu; r')$ of randomness -μ, where Com is a commitment algorithm and r' is an internal randomness, in advance. Suppose that the witness w is committed in the form of $\mathsf{c}_w := \mathsf{Com}(w; r)$. The prover also sends randomness $r^\star := r \odot_r r'$ along with $\sigma := w - \mu$, where $\odot_r$ is a homomorphic operation over the randomness space. Then, the verifier can check that:

$$\mathsf{Com}(\sigma; r^\star) \stackrel{?}{=} \mathsf{c}_w \odot_c \mathsf{c}_{-\mu} \ \ (= \mathsf{Com}(w + (-\mu); r + r')) \,, \tag{1}$$

where $\odot_c$ is a homomorphic operation over the commitment space. Although this procedure enables the verifier to confirm the equivalence of the prover's input and the committed value,

[1] To be precise, QuickSilver requires interactions only for multiplication gates where *both inputs are masked values*. No interactions are needed for multiplication gates with at least one constant input.

the verifier obtains σ, $\mathsf{c}_{-\mu}$, and $r^\star$ in addition to c_w to verify (1). Those values must be related to the original commitment c_w, and therefore, the standard hiding and binding properties are insufficient for the commitment scheme required in our range argument.

Hence, we define new hiding and binding properties, called *special hiding* and *special binding*, that guarantee that w remains hidden and unalterable, even if an adversary obtains σ, $\mathsf{c}_{-\mu}$, and $r^\star$. In section 4, we give formal definitions of special hiding and special binding, and we prove that any homomorphic commitment satisfies them. This flexibility enables our generic construction of the range proof, allowing our range argument not to rely on any particular mathematical assumptions. By adapting quantum-resistant commitment like [2], we can construct a post-quantum range argument.

Circuit Representation for Range Argument. To represent the constraints that the prover must prove in range arguments as arithmetic circuits, we pay attention to the bit-decomposition approach [5]. According to this approach, the prover must show the following constraints to prove $w \in [0, 2^n - 1]$:

- value-check: $w = \sum_{i=0}^{n-1} w_i \cdot 2^i$
- bit-check: $w_i \in \{0, 1\}$ for $i \in [0, n-1]$,

where $(w_0, \ldots, w_{n-1})$ is a bit representation of w. Note that both the value-check and the bit-check are necessary to verify whether $(w_0, \ldots, w_{n-1})$ represents a bit-wise decomposition of w, as the arithmetic circuit is defined over some field $\mathbb{F}_p$ of order p, and each w_i for $i \in [0, n-1]$ is an element of $\mathbb{F}_p$, not a binary value. Considering that $w_i \in \{0, 1\}$ $(\subset \mathbb{F}_p) \Leftrightarrow w_i(w_i - 1) = 0$ holds, we can convert the above constraints into the following functions:

- $f_{\text{val}}((w_0, \ldots, w_{n-1}, w)) \coloneqq \sum_{i=0}^{n-1} w_i \cdot 2^i - w$
- $f_{\text{bit-}i}(w_i) \coloneqq w_i(w_i - 1)$ for $i \in [0, n-1]$.

The prover can show that value-check and bit-check hold by proving $f_{\text{val}} = 0$ and each $f_{\text{bit-}i} = 0$. We can represent f_{val} and each $f_{\text{bit-}i}$ by only multiplication and addition operations. Then, we can represent constraints of range arguments by $n + 1$ arithmetic circuits.

3 Preliminaries

3.1 Notation

$x \xleftarrow{\$} S$ denotes a sampling of an element x from S uniformly at random. $\mathsf{out} \leftarrow \mathcal{A}(\mathsf{in})$ denotes that an algorithm $\mathcal{A}$ takes in as input and outputs out. For $a, b \in \mathbb{Z}$ with $a \leq b$, we write $[a, b] \coloneqq \{a, \ldots, b\}$. A bold lowercase letter $\boldsymbol{x} \coloneqq (x_1, \ldots, x_n)^\top$ denotes a column vector. A bold upper-case letter $\boldsymbol{A}$ denotes a matrix. A circuit $\mathcal{C}$ over a field $\mathbb{F}_p$ is defined by a set of input wires $\mathcal{I}_{\mathcal{C}}^{\text{in}}$ and output wires $\mathcal{I}_{\mathcal{C}}^{\text{out}}$, along with a list of gates of the form $(\alpha, \beta, \gamma, T)$, where α, β are the indices of the input wires of the gate, γ is the index of the output wire of the gate, and $T \in \{\mathsf{Add}, \mathsf{Mult}\}$ is the type of the gate. If $p > 2$ is prime, $\mathcal{C}$ is an arithmetic circuit where $\mathsf{Add}/\mathsf{Mult}$ correspond to addition/multiplication in $\mathbb{F}_p$. We write τ to denote the number of Mult gates in $\mathcal{C}$.

3.2 Zero-Knowledge Proof

A zero-knowledge proof of knowledge is a protocol in which a prover can convince a verifier that some statement holds without revealing any information about why it holds. An argument is

a proof which holds only if the prover is computationally bounded and certain computational hardness assumptions hold. We consider arguments consisting of three interactive probabilistic polynomial time (PPT) algorithms $(\mathsf{Setup}, \mathcal{P}, \mathcal{V})$. Setup is the common reference string (CRS) generator, $\mathcal{P}$ is the prover, and $\mathcal{V}$ is the verifier. Let $\mathcal{R} \subset \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ be a polynomial-time-decidable ternary relation. Given CRS σ, we call w a witness for a statement u if $(\sigma, u, w) \in \mathcal{R}$ and define the CRS-dependent language $\mathcal{L}_\sigma = \{u \mid \exists w \colon (\sigma, u, w) \in \mathcal{R}\}$ as the set of statements u that have a witness w in the relation $\mathcal{R}$. A zero-knowledge proof should satisfy three key properties:

- **Completeness**: A prover $\mathcal{P}$ succeeds in convincing a verifier $\mathcal{V}$ that a true statement as long as $u \in \mathcal{L}_\sigma$.
- **Soundness**: A malicious prover cannot convince a verifier $\mathcal{V}$ that a statement u is a true statement if $u \notin \mathcal{L}_\sigma$. We distinguish between computational soundness that protects against polynomial time cheating provers and statistical or perfect soundness where even an unbounded prover cannot convince the verifier of a false statement. We will call computationally sound proofs for arguments.
- **Zero-Knowledge**: A malicious verifier learns nothing except $u \in \mathcal{L}_\sigma$. We distinguish between computational zero-knowledge, where a polynomial time verifier learns nothing from the proof and statistical or perfect zero-knowledge, where even a verifier with unlimited resources learns nothing from the proof.

Due to the page limitation, we omit the precise definition as in [5], which defines completeness as perfect completeness, soundness as computational witness-extended emulation, and zero-knowledge as perfect special honest-verifier zero-knowledge (SHVZK).

3.3 VOLE and IT-MACs

We use vector oblivious linear evaluation (VOLE) and information-theoretic message authentication codes (IT-MACs) to construct our range argument. VOLE generates authenticated values, where $\mathcal{P}$ obtains $\boldsymbol{x} \in \mathbb{F}_p^\ell$ and $\boldsymbol{m} \in \mathbb{F}_p^\ell$, and $\mathcal{V}$ obtains $\Delta \in \mathbb{F}_p$ and $\boldsymbol{k} \in \mathbb{F}_p^\ell$ such that $\boldsymbol{m} = \boldsymbol{k} - \Delta \cdot \boldsymbol{x} \in \mathbb{F}_p^\ell$. We omit the definition of the functionality of VOLE $\mathcal{F}_{\mathsf{VOLE}}^p$ given in [19] as subfield VOLE. By using VOLE, we can construct IT-MACs [15] to authenticate values in a finite field $\mathbb{F}_p$. In more detail, let $\Delta \in \mathbb{F}_p$ be a global key that can be used repeatedly, sampled uniformly, and Δ is known only by one party $\mathcal{V}$. $x \in \mathbb{F}_p$ known by the other party $\mathcal{P}$ can be authenticated by giving $\mathcal{V}$ a uniform key $K[x] \in \mathbb{F}_p$ and giving $\mathcal{P}$ the corresponding MAC tag

$$M[x] = K[x] - \Delta \cdot x \in \mathbb{F}_p.$$

We denote such an authenticated value by $[x]$. We also denote the pair of authenticated values that $\mathcal{P}$ has as $[x]_\mathcal{P} = (x, M[x])$ and that $\mathcal{V}$ has as $[x]_\mathcal{V} = (\Delta, K[x])$. An authenticated value $[x]$ can be "opened" by having $\mathcal{P}$ send x and $M[x]$ to $\mathcal{V}$, who verifies $M[x] = K[x] - \Delta \cdot x$. This has a soundness error of $1/p$. $\mathcal{P}$ and $\mathcal{V}$ can obtain the authentication relationship for another value without communication by executing the following operations:

- **Homomorphic Addition**. $\mathcal{P}$ and $\mathcal{V}$ compute $[x''] = [x] + [x']$ from $[x]$ and $[x']$. That is, $\mathcal{P}$ computes $[x'']_\mathcal{P} \coloneqq (x + x', M[x] + M[x'])$. $\mathcal{V}$ computes $[x'']_\mathcal{V} \coloneqq (\Delta, K[x] + K[x'])$.
- **Constant Addition**. $\mathcal{P}$ and $\mathcal{V}$ compute $[y] = c + [x]$ from $[x]$ and a public constant c. That is, $\mathcal{P}$ computes $[y]_\mathcal{P} \coloneqq (c + x, M[x])$. $\mathcal{V}$ computes $[y]_\mathcal{V} \coloneqq (\Delta, K[x] + c\Delta)$.

- **Constant Multiplication**. $\mathcal{P}$ and $\mathcal{V}$ compute $[z] = c \cdot [x]$ from $[x]$ and a public constant c. That is, $\mathcal{P}$ computes $[z]_{\mathcal{P}} \coloneqq (cx, c \cdot M[x])$. $\mathcal{V}$ computes $[z]_{\mathcal{V}} \coloneqq (\Delta, c \cdot K[x])$.

Using $[x]$, $\mathcal{P}$ and $\mathcal{V}$ can obtain the another authenticated relation $[w]$ of the IT-MACs without revealing the value w possessed by $\mathcal{P}$ to $\mathcal{V}$. Specifically, $\mathcal{P}$ computes $\sigma \coloneqq w - x$, sends σ to $\mathcal{V}$, and obtains $[w]_{\mathcal{P}} = (w, M[w]) \coloneqq (w, M[x])$. On the other hand, $\mathcal{V}$ receives σ and obtain $[w]_{\mathcal{V}} = (\Delta, K[w]) \coloneqq (\Delta, K[x] + \Delta\sigma)$.

3.4 VOLE-based ZKP

ZKP-AC allows a prover $\mathcal{P}$ to convinces a verifier $\mathcal{V}$ that the prover knows a witness w for which $\mathcal{C}(w) = 0$ without leaking any extra information [13]. The detail of the definition of the functionality of ZKP-AC $\mathcal{F}_{\mathrm{ZK}}$ is given in [19]. In recent years, several ZKP-ACs based on subfield VOLE have been proposed [18, 19]. This types of ZKP-AC is called VOLE-based ZKP. VOLE-based ZKP has computationally efficient features such as splitting the protocol into a preprocessing phase and an online phase. In the preprocessing phase, $\mathcal{P}$ and $\mathcal{V}$ generate VOLE correlations. In the online phase, $\mathcal{P}$ generates the proofs based on multiplication and addition operations using the relation of IT-MACs. $\mathcal{V}$ checks the correctness of the output of an arithmetic circuit. Generating VOLE correlations in the preprocessing phase makes computation after fixing the input, that is, the online phase efficient. $\mathcal{P}$ must generate an additional proof that $\mathcal{P}$ multiplies correctly for every multiplication gate whose inputs are not both public. QuickSilver [19], one of VOLE-based ZKP, proposes the gate-by-gate paradigm to generate such proof efficiently.

Gate-by-Gate Paradigm. The gate-by-gate paradigm is a technique for proving the soundness of multiplication gates at once. For each multiplication gate $(\alpha, \beta, \gamma, \mathsf{Mult})$ in the arithmetic circuit $\mathcal{C}$, if $w_\gamma = w_\alpha \cdot w_\beta$ is satisfied, then, taking into account $m_i = k_i - \Delta \cdot w_i$, $i \in \{\alpha, \beta, \gamma\}$, the following relation must be satisfied:

$$
\begin{aligned}
&\overbrace{B_i \coloneqq k_\alpha \cdot k_\beta - k_\gamma \cdot \Delta}^{\mathcal{V} \text{ knows}} \\
&= m_\alpha \cdot m_\beta + (w_\beta \cdot m_\alpha + w_\alpha \cdot m_\beta - m_\gamma) \cdot \Delta + (w_\alpha \cdot w_\beta - w_\gamma) \cdot \Delta^2 \\
&= \underbrace{m_\alpha \cdot m_\beta}_{\coloneqq A_{0,i},\ \mathcal{P} \text{ knows}} + \underbrace{(w_\beta \cdot m_\alpha + w_\alpha \cdot m_\beta - m_\gamma)}_{\coloneqq A_{1,i},\ \mathcal{P} \text{ knows}} \cdot \underbrace{\Delta}_{\mathcal{V} \text{ knows}} .
\end{aligned}
$$

Moreover, when $\mathcal{C}$ has τ multiplication gates, the above relations can be checked at once using the $\mathcal{V}$'s challenge $\chi \xleftarrow{\$} \mathbb{F}_p$ as follows:

$$
\underbrace{\sum_{i \in [\tau]} B_i \cdot \chi^i}_{\coloneqq B(\mathcal{V} \text{ knows})} = \underbrace{\sum_{i \in [\tau]} A_{0,i} \cdot \chi^i}_{\coloneqq A_0(\mathcal{P} \text{ knows})} + \underbrace{\left(\sum_{i \in [\tau]} A_{1,i} \cdot \chi^i \right)}_{\coloneqq A_1(\mathcal{P} \text{ knows})} \cdot \underbrace{\Delta}_{\mathcal{V} \text{ knows}} .
$$

Furthermore, to satisfy the zero-knowledge property, $\mathcal{P}$ sends the proof $(U \coloneqq A_0 + M[\xi], V \coloneqq A_1 + \xi)$ to $\mathcal{V}$ using an additional VOLE $[\xi]$, and $\mathcal{V}$ verifies $B + K[\xi] = U + V \cdot \Delta$, thereby checking τ multiplication gates at once.

3.5 Commitment

A (non-interactive) commitment scheme is an essential building block for range arguments. It allows a sender to commit to a value x by sending a commitment c_x and then to reveal x by

opening the commitment later. A commitment scheme must satisfy the property of hiding, which requires the receiver not to obtain any information about x from the commitment that the sender has committed, and binding, which requires the sender not later to generate opening information that would change the value that the sender has already committed. That is, the value that the sender opens is guaranteed to be x. More precisely, we define them as follows:

Definition 1 (Commitment)**.** *A non-interactive commitment scheme π_{COM} consists of a pair of PPT algorithms (*CSetup, Com*).*

- $\mathsf{pp} \leftarrow \mathsf{CSetup}(1^\kappa)$*: The setup algorithm* CSetup *is a PPT algorithm that takes security parameter κ as input and outputs public parameters* pp *for the scheme. Message space $\mathcal{X}$, randomness space $\mathcal{R}$ and commitment space $\mathcal{C}$ are defined by* pp*.*
- $\mathsf{c}_x \leftarrow \mathsf{Com}_{\mathsf{pp}}(x)$*: The commitment algorithm* $\mathsf{Com}_{\mathsf{pp}}$ *is a PPT algorithm (linked to* pp*) that takes a message $x \in \mathcal{X}$ as input, draws $r \xleftarrow{\$} \mathcal{R}$ uniformly at random, and computes commitment $\mathsf{c}_x \coloneqq \mathsf{Com}_{\mathsf{pp}}(x; r)$. For simplicity, we will write* $\mathsf{Com}_{\mathsf{pp}}$ *as* Com*.*

Definition 2 (Hiding)**.** *A non-interactive commitment scheme π_{COM} is said to be hiding if for all PPT adversaries $\mathcal{A}$ and sufficiently large security parameter κ, there exists a negligible function* $\mathsf{negl}(\kappa)$ *such that*

$$\left| \Pr\left[\begin{array}{l} \mathsf{pp} \leftarrow \mathsf{CSetup}(1^\kappa),\ (x_0, x_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}), \\ b \xleftarrow{\$} \{0,1\},\ r \xleftarrow{\$} \mathcal{R},\ \mathsf{c}_{x_b} \coloneqq \mathsf{Com}(x_b; r), \\ b' \leftarrow \mathcal{A}(\mathsf{st}, \mathsf{c}_{x_b}) \end{array} : b' = b \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\kappa).$$

where the probability is over b, r, CSetup *and $\mathcal{A}$. Especially, if* $\mathsf{negl}(\kappa) = 0$ *then we say the π_{COM} is perfectly hiding.*

Definition 3 (Binding)**.** *A non-interactive commitment scheme π_{COM} is said to be binding if for all PPT adversaries $\mathcal{A}$ and sufficiently large security parameter κ, there exists a negligible function* $\mathsf{negl}(\kappa)$ *such that*

$$\Pr\left[\begin{array}{l} \mathsf{pp} \leftarrow \mathsf{CSetup}(1^\kappa), \\ (x_0, x_1, r_0, r_1) \leftarrow \mathcal{A}(\mathsf{pp}) \end{array} : \ \mathsf{c}_{x_0} = \mathsf{c}_{x_1} \wedge\ x_0 \neq x_1 \right] \leq \mathsf{negl}(\kappa)$$

where $\mathsf{c}_{x_i} \coloneqq \mathsf{Com}(x_i; r_i)$ $(i \in \{0,1\})$ and the probability is over CSetup *and $\mathcal{A}$. Especially, If* $\mathsf{negl}(\kappa) = 0$ *then we say the π_{COM} is perfectly binding.*

In this paper, we consider homomorphic commitments.

Definition 4 (Homomorphic Commitment)**.** *A homomorphic commitment scheme is a non-interactive commitment scheme such that $(\mathcal{X}, \odot_x)$, $(\mathcal{R}, \odot_r)$ and $(\mathcal{C}, \odot_c)$ are all abelian groups, and for all $x_0, x_1 \in \mathcal{X}$, $r_0, r_1 \in \mathcal{R}$, we have*

$$\mathsf{Com}(x_0; r_0) \odot_c \mathsf{Com}(x_1; r_1) = \mathsf{Com}(x_0 \odot_x x_1; r_0 \odot_r r_1).$$

4 New Security Requirement for Commitment

As mentioned in section 2, we need to force the prover to bind the message w used in the commitment c_w for the range argument to the witness w used in $\sigma \coloneqq w - \mu$. We define special binding, which guarantees that w in σ is bound to w in c_w; both w are the same. On the other hand, in the context of commitment, the verifier needs to obtain σ, c_μ, and $r^\star$ in addition to c_w to verify (1). We also define special hiding, which guarantees that w remains hidden even if the verifier obtains σ, c_μ, and $r^\star$. Precisely, we define special binding and special hiding.

Definition 5 (Special Binding)**.** *A homomorphic commitment scheme π_{COM} satisfies special binding if for all PPT adversaries $\mathcal{A}$ and sufficiently large security parameter κ, there exists a negligible function $\mathsf{negl}(\kappa)$ such that*

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathsf{CSetup}(1^\kappa),\ \mu \xleftarrow{\$} \mathcal{X},\\ (x_0, x_1, r_0, r_1, r) \leftarrow \mathcal{A}(\mathsf{pp}, \mu), \\ \mathsf{c}_0 \coloneqq \mathsf{c}_{x_0} \odot_c \mathsf{c}_\mu,\ \mathsf{c}_1 \coloneqq \mathsf{c}_{x_1} \odot_c \mathsf{c}_\mu\end{array} : \ \mathsf{c}_0 = \mathsf{c}_1 \wedge\ x_0 \neq x_1\right] \leq \mathsf{negl}(\kappa),$$

where $\mathsf{c}_{x_i} \coloneqq \mathsf{Com}(x_i; r_i)$ $(i \in \{0,1\})$ and $\mathsf{c}_\mu \coloneqq \mathsf{Com}(\mu; r)$. Especially, if $\mathsf{negl}(\kappa) = 0$, then we say the π_{COM} is perfectly special binding.

Even if $\mathsf{c}_{x_0} \neq \mathsf{c}_{x_1}$, the adversary wins the game when the homomorphic operation results in $\mathsf{c}_0 = \mathsf{c}_1$ for a specific value μ.

Definition 6 (Special Hiding)**.** *A homomorphic commitment scheme π_{COM} satisfies special hiding if for all PPT adversaries $\mathcal{A}$ and sufficiently large security parameter κ, there exists a negligible function $\mathsf{negl}(\kappa)$ such that*

$$\left|\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathsf{CSetup}(1^\kappa),\ (x_0, x_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}),\\ b \xleftarrow{\$} \{0,1\},\ r \xleftarrow{\$} \mathcal{R},\ \mathsf{c}_{x_b} \coloneqq \mathsf{Com}(x_b; r),\\ x' \xleftarrow{\$} \mathcal{X},\ r' \xleftarrow{\$} \mathcal{R},\ \mathsf{c}_{x'} \coloneqq \mathsf{Com}(x'; r'),\\ x^\star \coloneqq x_b \odot_x x',\ r^\star \coloneqq r \odot_r r',\\ b' \leftarrow \mathcal{A}(\mathsf{st}, \mathsf{c}_{x_b}, \mathsf{c}_{x'}, x^\star, r^\star)\end{array} : b' = b\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\kappa).$$

In particular, if $\mathsf{negl}(\kappa) = 0$ for any κ, then π_{COM} satisfies perfectly special hiding.

In other words, even if an adversary obtains not only the challenging commitment c_{x_b} but also the commitment $\mathsf{c}_{x'}$ to the uniformly random number x', the operation result between the committed values $x^\star$, and the operation result between the random numbers used to generate each commitment $r^\star$, we can guarantee that the hiding property of the challenge x_b is guaranteed.

Theorem 1 (Special Binding)**.** *Any homomorphic commitment scheme satisfies special binding with the same advantage.*

Proof. Let A be an adversary against the special binding of a homomorphic commitment scheme π_{COM}, and A' be an adversary against the binding of a (homomorphic) commitment scheme. We construct A' by using any arbitrarily fixed A. The adversary A' of hiding behaves as the challenger of the special binding game. Let C be the challenger of the binding game. The challenger A' interacts A as follows:

1. C obtains $\mathsf{pp} \xleftarrow{\$} \mathsf{CSetup}$ and sends pp to A'.
2. A' chooses $\mu \in \mathcal{X}$ and sends pp and μ to A.
3. A' receives a challenge query (x_0, x_1, r_0, r_1, r) from A.

Suppose that the challenge query (x_0, x_1, r_0, r_1, r) from A meets special binding That is, (x_0, x_1, r_0, r_1, r) meets $\mathsf{c}_0 = \mathsf{c}_1$ and $x_0 \neq x_1$. C computes $\mathsf{c}_{x_0} \coloneqq \mathsf{Com}(x_0; r_0)$ and $\mathsf{c}_{x_1} \coloneqq \mathsf{Com}(x_1; r_1)$. The commitments are group elements, then the following equation holds.

$$\mathsf{c}_0 = \mathsf{c}_1 \quad \Leftrightarrow \quad \mathsf{c}_{x_0} \odot_c \mathsf{c}_\mu = \mathsf{c}_{x_1} \odot_c \mathsf{c}_\mu$$

$$\Leftrightarrow \quad \mathsf{c}_{x_0} \odot_c \mathsf{c}_\mu \odot_c \mathsf{c}_\mu^{-1} = \mathsf{c}_{x_1} \odot_c \mathsf{c}_\mu \odot_c \mathsf{c}_\mu^{-1}$$
$$\Leftrightarrow \quad \mathsf{c}_{x_0} = \mathsf{c}_{x_1}.$$

If A wins the game of special binding, A' wins the game of binding with the same advantage. □

Theorem 2 (Special Hiding)**.** *Any homomorphic commitment scheme satisfies special hiding with the same advantage.*

Proof. Let A be an adversary against the special hiding of a homomorphic commitment scheme π_{COM}, and A' be an adversary against the hiding of a (homomorphic) commitment scheme. We construct A' by using any arbitrarily fixed A. The adversary A' of hiding behaves as the challenger of the special hiding game. Let C be the challenger of the hiding game. At the beginning of the game, A' receives a public parameter pp and forwards it to A. The challenger A' interacts A as follows:

1. A' receives a challenge query (x_0, x_1) from A and forwards it to the challenger C. C samples $b \xleftarrow{\$} \{0,1\}$, $r \xleftarrow{\$} \mathcal{R}$, computes $\mathsf{c}_{x_b} \coloneqq \mathsf{Com}(x_b; r)$, and send it to $\mathcal{A}'$.
2. A' samples $x^* \xleftarrow{\$} \mathcal{X}$ and $r^* \xleftarrow{\$} \mathcal{R}$ and computes $\mathsf{c}_{x'} \coloneqq \mathsf{Com}(x^\star; r^\star) \odot_c \mathsf{c}_{x_b}^{-1}$. A' send $(\mathsf{c}_{x_b}, \mathsf{c}_{x'}, x^\star, r^\star)$ to A.
3. A predict $b' \in \{0,1\}$ and sends b' to A'
4. A' receives b' and forward b' to C.

$x^\star$ and $r^\star$ are independent of b. b is included in c_{x_b} and in $\mathsf{c}_{x_b}^{-1}$ in $\mathsf{c}_{x'}$. Then, if A wins the game of special hiding, A' wins the game of hiding with the same advantage. □

Therefore, any homomorphic commitment satisfies special binding and special hiding with the same advantage, which makes our range argument a generic construction that does not rely on any particular mathematical assumptions.

5 Our Range Argument Construction

5.1 Range Arguments

Range arguments aim to prove that a prover's committed value falls within a specified range for a verifier. Precisely, range arguments are zero-knowledge arguments for the following NP language $\mathcal{L}$:

$$\mathcal{L} = \left\{ (\mathsf{pp}, \mathsf{c}_w, n) \;\middle|\; \begin{array}{l} \exists (w, r) \in \mathcal{X} \times \mathcal{R} \\ \text{s.t. } \mathsf{c}_w = \mathsf{Com}(w; r) \;\wedge\; w \in [0, 2^n - 1] \end{array} \right\}.$$

This is the bit-decomposition-based definition and this definition generalizes to arbitrary intervals $[a, b]$. Our protocol is based on bit-decomposition so that we will give an overview.

Bit-Decomposition Approach. Bit-decomposition takes an approach based on the premise that a value within a certain range can be represented by a bit string representing that range. We consider the proof of $w \in [0, 2^n - 1]$. At first, the prover commits w by $\mathsf{c}_w = \mathsf{Com}(w; r)$. In the proof generation, the prover converts w into a bit representation $(w_0, \ldots, w_{n-1})$ and proves the following two constraints hold, without leaking knowledge of w or each w_i:

- value-check: The original w can be restored from w_i for $i \in [0, n-1]$. In other words, $w = \sum_{i=0}^{n-1} w_i \cdot 2^i$ holds.
- bit-check: each w_i for $i \in [0, n-1]$ is a bit. In other words, $w_i \in \{0, 1\}$ holds for $i \in [0, n-1]$.

One of the state-of-the-art methods in this paradigm is Bulletproofs [5], which features very small proof size $\mathcal{O}(\kappa \cdot \log n)$ for a security parameter κ and enjoys a transparent setup. Therefore, Bulletproofs have become the most commonly used solution in real-world applications.

5.2 Conversion to Circuit Representation

We show that we can convert constraints of value-check and bit-check into arithmetic circuits to adapt QuickSilver [19].

Conversion for Value-Check. In order to convert $w = \sum_{i=0}^{n-1} w_i \cdot 2^i$ into an arithmetic circuit representation, we define

$$f_{\text{val}}((w_0, \ldots, w_{n-1}, w)) \coloneqq \sum_{i=0}^{n-1} w_i \cdot 2^i - w.$$

We can express f_{val} by addition and multiplication operations. Therefore, we can convert f_{val} into an arithmetic circuit $\mathcal{C}_{\text{val}}$ as shown in Figure 1. The prover can prove that $w = \sum_{i=0}^{n-1} w_i \cdot 2^i$ holds if he can show $\mathcal{C}_{\text{val}}((w_0, \ldots, w_{n-1}, w)) = 0$ holds.

Conversion for Bit-Check. Next, we will convert $w_i \in \{0, 1\}$ for $i \in [0, n-1]$ into arithmetic circuits representation. We note that for any $i \in [0, n-1]$, if $w_i(w_i - 1) = 0$, then $w_i \in \{0, 1\}$ holds identically. We define

$$f_{\text{bit-}i}(w_i) \coloneqq w_i(w_i - 1).$$

We can express $f_{\text{bit-}i}$ by addition and multiplication operations. Therefore, we can convert $f_{\text{bit-}i}$ into each arithmetic circuit $\mathcal{C}_{\text{bit-}i}$ as shown in Figure 2. If $\mathcal{C}_{\text{bit-}i}(w_i) = 0$ for $i \in [0, n-1]$, then we can prove that $w_i \in \{0, 1\}$ for $i \in [0, n-1]$ holds. $\mathcal{V}$ can be confident that the prover holds $w \in [0, 2^n - 1]$ by verifying that the outputs of $\mathcal{C}_{\text{val}}$ and each $\mathcal{C}_{\text{bit-}i}$ are all 0.

Since we have already converted value-check and bit-check into $n+1$ arithmetic circuits $\mathcal{C}_{\text{val}}$ and $\mathcal{C}_{\text{bit-}i}$ for $i \in [0, n-1]$, we can construct our range argument by simply executing QuickSilver $n+1$ times for $n+1$ arithmetic circuits.

Combining Verification of Gate-by-Gate Paradigm. In QuickSilver, $\mathcal{P}$ and $\mathcal{V}$ generate VOLE correlations in the preprocessing phase, which makes computation after fixing the input efficient. However, reducing the number of VOLE correlations is desirable in terms of communication and computational efficiency. We can reduce the number of VOLE correlations required for the gate-by-gate paradigm for $\mathcal{C}_{\text{bit-}i}$ for $i \in [0, n-1]$ from n to 1. We combine the n times gate-by-gate paradigm for n arithmetic circuits $\mathcal{C}_{\text{bit-}i}$ for $i \in [0, n-1]$ into one. Specifically, $\mathcal{V}$ samples $\chi \xleftarrow{\$} \mathbb{F}_p$ as a challenge, and send χ to $\mathcal{P}$. For each multiplication gate i that each $\{\mathcal{C}_{\text{bit-}i}\}_{i \in [0,n-1]}$ has, $\mathcal{P}$ computes $U \coloneqq \sum_{i \in [0,n-1]} A_{0,i} \cdot \chi^i + M[\xi]$ and $V \coloneqq \sum_{i \in [0,n-1]} A_{1,i} \cdot \chi^i + \xi$, and send (U, V) to $\mathcal{V}$. $\mathcal{V}$ computes $W \coloneqq \sum_{i \in [0,n-1]} B_i \cdot \chi^i + K[\xi]$. $\mathcal{V}$ can verify n multiplication gates at once by $W = U + \Delta \cdot V$.

By reconsidering the necessary circuits based on the above discussion, $n+1$ arithmetic circuits $\{\mathcal{C}_{\text{bit-}i}\}_{i \in [0,n-1]}$ and $\mathcal{C}_{\text{val}}$ can be expressed as a single arithmetic circuit $\mathcal{C}_{\text{rp}}$ with $n+1$ inputs and $n+1$ outputs in Figure 3. By running QuickSilver on $\mathcal{C}_{\text{rp}}$, we can execute value-check and bit-check in one go.

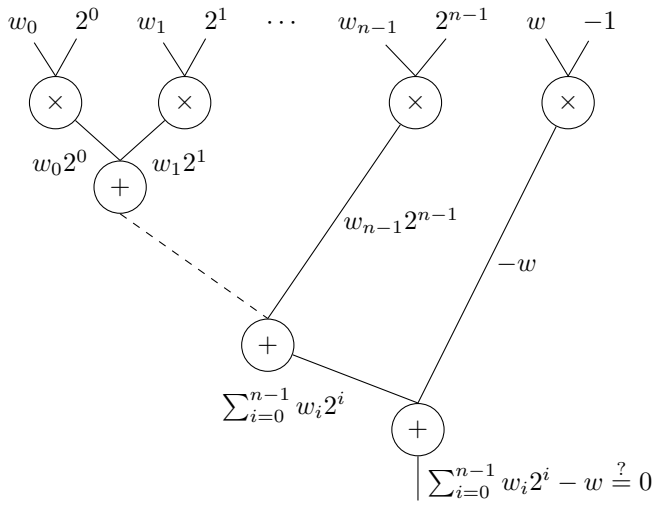

Figure 1: The arithmetic circuit $\mathcal{C}_{\text{val}}$.

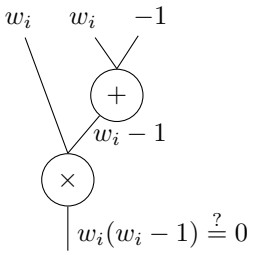

Figure 2: The arithmetic circuit $\mathcal{C}_{\text{bit-}i}$.

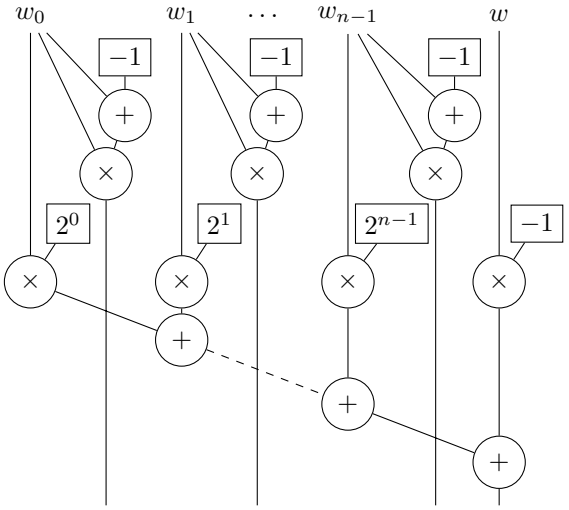

Figure 3: The arithmetic circuit $\mathcal{C}_{\text{rp}}$ for range arguments.

5.3 Our Range Argument Protocol

We propose a range argument protocol Π^p_{RA} in Figure 4 and 5. We use VOLE $\mathcal{F}^p_{\mathsf{VOLE}}$ and homomorphic commitment π_{COM}. In the preprocessing phase, $\mathcal{P}$ and $\mathcal{V}$ use functionality $\mathcal{F}^p_{\mathsf{VOLE}}$ to obtain $4n + 2$ VOLE correlations and $\mathcal{P}$ commits one VOLE randomness μ by $\mathsf{c}_{-\mu}$. $\mathcal{P}$ commits the witness w by c_w. In the online phase, $\mathcal{P}$ and $\mathcal{V}$ obtain authenticated values of the each witness w_i by using $\sigma_i := w_i - \mu_i$. $\mathcal{P}$ sends $\sigma_n := w_n - \mu$ and $r^\star := r + r'$ to $\mathcal{V}$, and $\mathcal{V}$ verifies that w in σ is bound to w in c_w by $\mathsf{Com}(\sigma_n; r^\star) = \mathsf{c}_w \odot_c \mathsf{c}_{-\mu}$. Then, $\mathcal{P}$ generates zero-knowledge proofs for the arithmetic circuit $\mathcal{C}_{\text{rp}}$ using VOLE and IT-MAC. $\mathcal{V}$ verifies that

the proof (U, V) for gate-by-gate paradigm satisfies $W = U + \Delta \cdot V$, and the proof $M[w_{h_i}]$ corresponding to each output wires h_i for $i \in [0, n-1]$ satisfies $M[w_{h_i}] = K[w_{h_i}]$. After all checks are passed, the verifier can be convinced of $w \in [0, 2^n - 1]$.

Theorem 3. *The range arguments Π^p_{RA} have perfect completeness, computational witness-extended emulation, and SHVZK with soundness error $(n+3)/p + \epsilon$, where ϵ is the advantage of binding of π_{COM}.*

Due to the page limitation, we focus on the effect caused by incorporating a commitment π_{COM} into QuickSilver and omit full proof. The adversary $\mathcal{A}$ wants to forge $w' \neq w$. Let $w' \coloneqq w + e$, where e is an error. If $\mathcal{A}$ uses a correct VOLE randomness μ, $\mathcal{A}$ sends $\sigma'_n \coloneqq w' - \mu$ to the verifier. In this case, $\mathcal{A}$ needs to break special binding, that is, $\mathcal{A}$ needs to find σ'_n such that $\mathsf{Com}(\sigma'_n; r^\star) = \mathsf{c}_w \odot_c \mathsf{c}_{-\mu}$ holds. This advantage is ϵ. Else if $\mathcal{A}$ uses an incorrect VOLE randomness μ', $\mathcal{A}$ sends $\sigma'_n \coloneqq w' - \mu'$ to the verifier. By setting $\mu' = \mu - e$, which makes $\sigma'_n = \sigma_n$, the verifier $\mathcal{V}$ will pass the commitment check $\mathsf{Com}(\sigma'_n; r^\star) = \mathsf{c}_w \odot_c \mathsf{c}_{-\mu}$, but does not pass the subsequent IT-MAC verification. $\mathcal{A}$ sets $M[w'] \coloneqq M[\mu]$. $\mathcal{V}$ computes $K[w'] \coloneqq K[\mu] + \Delta\sigma'_n = K[\mu] + \Delta(w' - \mu) + \Delta e$. Here, the adversary $\mathcal{A}$ must satisfy $M[w'] = K[w'] - \Delta w' \Leftrightarrow M[\mu] = K[\mu] - \Delta\mu + \Delta e$. The probability that $\mathcal{A}$ can estimate Δ such that $\Delta e = 0$ is $1/p$ and the probability that $\mathcal{A}$ can estimate Δ that can pass the gate-by-gate paradigm verification is $2/p$.

6 Experimental Evaluation

We conducted performance evaluations to benchmark our range argument against Bulletproofs [5], which is one of the state-of-the-art bit-decomposition constructions. Bulletproofs have become the most commonly used solution in real-world applications, and other state-of-the-art range proofs generally compare Bulletproofs, so it is also suitable for comparison with our range argument. The other mainstream is the square-decomposition approach, but it has the drawback of relaxed soundness and need RSA or large class groups to address the relaxed soundness. Then, we have not evaluated square-decomposition constructions like Sharp [8].

Protocol Π^p_{RA} (Part I)

Inputs: The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ hold a circuits $\mathcal{C}_{\text{rp}}$ over $\mathbb{F}_p$, which has $n+1$ inputs and $n+1$ outputs. The prover $\mathcal{P}$ also holds a witness w and its bit representation $w_0, w_1, \ldots, w_{n-1}$ such that $\mathcal{C}_{\text{rp}}((w_0, w_1, \ldots, w_{n-1}, w)) = \mathbf{0}^{n+1}$.

Preprocessing Phase: We denote μ_{3n} as μ.

1. $\mathcal{P}$ and $\mathcal{V}$ send (init) to $\mathcal{F}^p_{\mathsf{VOLE}}$, witch returns a uniform $\Delta \in \mathbb{F}_p$ to $\mathcal{V}$.
2. $\mathcal{P}$ and $\mathcal{V}$ send $(\mathsf{extend}, 4n+2)$ to $\mathcal{F}^p_{\mathsf{VOLE}}$, which returns $\{[\mu_i]_{\mathcal{P}} \coloneqq (\mu_i, M[\mu_i])\}_{i\in[0,3n]}$, $\{[\nu_i]_{\mathcal{P}} \coloneqq (\nu_i, M[\nu_i])\}_{i\in[0,n-1]}$, $[\xi]_{\mathcal{P}} \coloneqq (\xi, M[\xi])$ to $\mathcal{P}$ and $\{[\mu_i]_{\mathcal{V}} \coloneqq (\Delta, K[\mu_i])\}_{i\in[0,3n]}$, $\{[\nu_i]_{\mathcal{V}} \coloneqq (\Delta, K[\nu_i])\}_{i\in[0,n-1]}$, $[\xi]_{\mathcal{V}} \coloneqq (\Delta, K[\xi])$ to $\mathcal{V}$.
3. $\mathcal{P}$ generates $r' \xleftarrow{\$} \mathcal{R}$ $(= \mathbb{F}_p)$, computes $\mathsf{c}_{-\mu} \coloneqq \mathsf{Com}(-\mu; r')$ and sends $\mathsf{c}_{-\mu}$ to $\mathcal{V}$.

Commit Phase: $\mathcal{P}$ computes $\mathsf{c}_w \coloneqq \mathsf{Com}(w; r)$ and publishes it.

Figure 4: Our range argument over any field (Part I).

Protocol Π^p_{RA} (Part II)

Online Phase: We denote w as w_n. Let (w_α, w_β) be the values on the left and right input wires (α, β). For $i \in \mathcal{I}^{\mathrm{in}}_{\mathcal{C}_{\mathrm{rp}}} \setminus \{n\} (= [0, n-1])$, let $w_i^{(\ell)}$ for $\ell \in \{0,1,2\}$ be values on each wires from input value w_i.

1. For $i \in [0, n-1]$, $\mathcal{P}$ sends values $\sigma_i^{(\ell)} \coloneqq w_i^{(\ell)} - \mu_{\ell(n-1)+i}$ for $\ell \in \{0,1,2\}$ and $\sigma_n \coloneqq w_n - \mu$ to $\mathcal{V}$. $\mathcal{P}$ computes $[w_i^{(\ell)}]_{\mathcal{P}} = (w_i^{(\ell)}, M[w_i^{(\ell)}]) \coloneqq (w_i^{(\ell)}, M[\mu_{\ell n+i}])$ and $[w_n]_{\mathcal{P}} = (w_n, M[w_n]) \coloneqq (w_n, M[\mu])$. $\mathcal{V}$ also computes $[w_i^{(\ell)}]_{\mathcal{V}} = (\Delta, K[w_i^{(\ell)}]) \coloneqq (\Delta, K[\mu_{\ell n+i}] + \Delta\sigma_i)$ and $[w_n]_{\mathcal{V}} = (\Delta, K[w_n]) \coloneqq (\Delta, K[\mu] + \Delta\sigma_n)$.
2. $\mathcal{P}$ computes $r^\star \coloneqq r + r'$ and sends $r^\star$. $\mathcal{V}$ checks $\mathsf{Com}(\sigma_n; r^\star) \stackrel{?}{=} \mathsf{c}_w \odot_c \mathsf{c}_{-\mu}$. If the check fails, then $\mathcal{V}$ outputs false.
3. For each gate $(\alpha, \beta, \gamma, T) \in \mathcal{C}_{\mathrm{rp}}$, execute the following in the topological order, where $i \in [0, n-1]$. For simplicity, We denote $[w_i^{(\ell)}]_{\mathcal{P}}$ (resp. $[w_i^{(\ell)}]_{\mathcal{V}}$) for each $\ell \in \{0,1,2\}$ as $[w_i]_{\mathcal{P}} = (w_i, M[w_i])$ (resp. $[w_i]_{\mathcal{V}} = (\Delta, K[w_i])$) below:
   - Case $T = \mathsf{Add}$ and $(w_\alpha, w_\beta) = (w_i, -1)$: Since one of the inputs of the Add gate is the public constant -1, $\mathcal{P}$ computes $[w_i - 1]_{\mathcal{P}} = (w_i - 1, M[w_i - 1]) \coloneqq (w_i - 1, M[w_i])$, and $\mathcal{V}$ computes $[w_i - 1]_{\mathcal{V}} = (\Delta, K[w_i - 1]) \coloneqq (\Delta, K[w_i] - \Delta)$, without communicating with each other.
   - Case $T = \mathsf{Mult}$ and $(w_\alpha, w_\beta) = (w_i, w_i - 1)$: Both inputs w_α and w_β of the Mult gate are not public. $\mathcal{P}$ computes $d_i \coloneqq w_i \cdot (w_i - 1) - \nu_i \in \mathbb{F}_p$ for $i \in [0, n-1]$ and send them to $\mathcal{V}$. $\mathcal{V}$ computes $[w_i(w_i - 1)]_{\mathcal{V}} = (\Delta, K[w_i(w_i - 1)]) \coloneqq (\Delta, K[\nu_i] + \Delta d_i)$ for $i \in [0, n-1]$.
   - Case $T = \mathsf{Mult}$ and $(w_\alpha, w_\beta) = (w_i, 2^i)$: Since one of the inputs of the Mult gate is the public constant 2^i, $\mathcal{P}$ computes $[2^i w_i]_{\mathcal{P}} = (2^i w_i, M[2^i w_i]) \coloneqq (2^i w_i, 2^i \cdot M[w_i])$ and $\mathcal{V}$ computes $[2^i w_i]_{\mathcal{V}} = (\Delta, K[2^i w_i]) \coloneqq (\Delta, 2^i \cdot K[w_i])$ without communicating with each other.
   - Case $T = \mathsf{Add}$ and $w_\alpha = w_i 2^i$ or $w_\beta = w_n$: Although both inputs w_α and w_β to the Add gate are not public, for using the additive homomorphism of IT-MACs, $\mathcal{P}$ computes $[w_\gamma]_{\mathcal{P}} \coloneqq (w_\alpha + w_\beta, M[w_\alpha] + M[w_\beta])$ and $\mathcal{V}$ computes $[w_\gamma]_{\mathcal{V}} \coloneqq (\Delta, K[w_\alpha] + K[w_\beta])$ without communicating with each other.
4. For each Mult gate $i \in [0, n-1]$ which inputs (w_α, w_β) are $(w_i, w_i - 1)$, $\mathcal{P}$ has $([w_\gamma]_{\mathcal{P}}, [w_\alpha]_{\mathcal{P}}, [w_\beta]_{\mathcal{P}})$ and $\mathcal{V}$ also has $([w_\gamma]_{\mathcal{V}}, [w_\alpha]_{\mathcal{V}}, [w_\beta]_{\mathcal{V}})$ as an authenticated triple $(m_i = k_i - \Delta \cdot w_i, i \in \{\alpha, \beta, \gamma\})$. To verify that $\mathcal{P}$ has correctly computed $w_\gamma = w_\alpha \cdot w_\beta$ for each multiplication gate $i \in [0, n-1]$ all at once, both parties execute the following.
   (a) For $i \in [0, n-1]$, $\mathcal{P}$ computes $A_{0,i} \coloneqq m_\alpha \cdot m_\beta \in \mathbb{F}_p$ and $A_{1,i} \coloneqq w_\alpha \cdot m_\beta + w_\beta \cdot m_\alpha - m_\gamma \in \mathbb{F}_p$, and $\mathcal{V}$ also computes $B_i \coloneqq k_\alpha \cdot k_\beta - k_\gamma \cdot \Delta \in \mathbb{F}_p^r$.
   (b) $\mathcal{V}$ generates $\chi \xleftarrow{\$} \mathbb{F}_p$ and send χ to $\mathcal{P}$.
   (c) $\mathcal{P}$ computes $U \coloneqq \sum_{i \in [0,n-1]} A_{0,i} \cdot \chi^i + M[\xi]$ and $V \coloneqq \sum_{i \in [0,n-1]} A_{1,i} \cdot \chi^i + \xi$ using $A_{0,i}, A_{1,i}$. $\mathcal{V}$ computes $W \coloneqq \sum_{i \in [0,n-1]} B_i \cdot \chi^i + K[\xi]$ using B_i.
5. $\mathcal{P}$ sends (U, V) and m_{h_i} for all $h_i \in \mathcal{I}^{\mathrm{out}}_{\mathcal{C}_{\mathrm{rp}}}$ to $\mathcal{V}$. $\mathcal{V}$ checks $W \stackrel{?}{=} U + \Delta V$ and $K[w_{h_i}] \stackrel{?}{=} M[w_{h_i}]$ for all $h_i \in \mathcal{I}^{\mathrm{out}}_{\mathcal{C}_{\mathrm{rp}}}$. If the checks fail, $\mathcal{V}$ outputs false. Otherwise, $\mathcal{V}$ outputs true.

Figure 5: Our range argument over any field (Part II).

Table 1: Compering computational costs of our range arguments with Bulletproofs over a field $\mathbb{F}_p$. All timings are in milliseconds.

$p \approx 2^{128}$	$n = 32$		$n = 64$	
	$\mathcal{P}$	$\mathcal{V}$	$\mathcal{P}$	$\mathcal{V}$
Bulletproofs [5]	3.88	0.60	7.38	1.04
the online phase of $\Pi^p_{\text{RA-Ped}}$	0.51	1.35	0.63	1.42

The benchmarks were made on a laptop with an Apple M2 Max, 64GB RAM, running macOS 14. We implement our scheme using C++, based on an existing implementation of QuickSilver[1]. In detail, we use the P-256 curve for the Pedersen commitment [16] as a widely standard commitment scheme and generate VOLE correlations using [18] without any VOLE extensions. We call this range argument scheme $\Pi^p_{\text{RA-Ped}}$. We measured the running time of only the online phase of $\Pi^p_{\text{RA-Ped}}$ in terms of proving and verification time, respectively. We also implement Bulletproof using an existing implementation[2]. The implementation result of computational costs is in Table 1. The total computation time for the prover and verifier in the online phase is more efficient than that of Bulletproofs. Especially, the prover computation is efficient. The verifier computation is not as fast as Bulletproofs, but it is still efficient enough. Since no witness information is communicated during the preprocessing phase, the prover and verifier can execute the preprocessing phase computation in advance. Therefore, our range argument is efficient in the online phase. Including the computations of the preprocessing phase, it takes about 25-27 ms for 32-bit and 64-bit, but our range argument is sufficiently practical. Moreover, our range argument is a generic construction. By adapting BDLOP commitment [2] as quantum-resistant commitment, we can construct a post-quantum range argument. We also benchmark the commit phase of BDLOP commitment itself, and it takes 78.3(ms). It is dominant for our range argument, but it will still be efficient.

7 Conclusion

We proposed an efficient range arguments protocol with preprocessing phase. The main idea is to express the constraints that the prover must prove in range arguments as arithmetic circuits. To adapt the constraints to ZKP-AC, we define new hiding and binding properties, called special hiding and special binding, and we prove that any homomorphic commitment satisfies them with the same advantage, which makes our range argument a generic construction that does not rely on any particular mathematical assumptions, enabling us to construct a post-quantum range argument. The implementation evaluation showed that the total computation time for the prover and verifier in the online phase is efficient compared to Bulletproofs, one of the state-of-the-art methods. Especially, the prover computation is efficient.

References

[1] Olivier Baudron, Pierre-Alain Fouque, David Pointcheval, Jacques Stern, and Guillaume Poupard. Practical multi-candidate election system. In *PODC*, pages 274–283, 2001.

[1] https://github.com/emp-toolkit
[2] https://github.com/dalek-cryptography/bulletproofs/

[2] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In *SCN 2018*, pages 368–385, 2018.

[3] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT*, pages 431–444, 2000.

[4] Benedikt Bünz, Shashank Agrawal, Mahdi Zamani, and Dan Boneh. Zether: Towards privacy in a smart contract world. In *FC*, pages 423–443, 2020.

[5] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *SP*, pages 315–334, 2018.

[6] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT*, pages 302–321, 2005.

[7] Konstantinos Chalkias, Shir Cohen, Kevin Lewi, Fredric Moezinia, and Yolan Romailler. Hashwires: Hyperefficient credential-based range proofs. *PETS*, pages 76–95, 2021.

[8] Geoffroy Couteau, Dahmun Goudarzi, Michael Klooß, and Michael Reichle. Sharp: Short relaxed range proofs. In *CCS*, pages 609–622, 2022.

[9] Geoffroy Couteau, Michael Klooß, Huang Lin, and Michael Reichle. Efficient range proofs with transparent setup from bounded integer commitments. In *EUROCRYPT*, pages 247–277, 2021.

[10] Geoffroy Couteau, Thomas Peters, and David Pointcheval. Removing the strong RSA assumption from arguments over the integers. In *EUROCRYPT*, pages 321–350, 2017.

[11] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

[12] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *STOC*, pages 291–304, 1985.

[13] Jens Groth. On the size of pairing-based non-interactive arguments. In *EUROCRYPT*, pages 305–326, 2016.

[14] Gregory Maxwell. Confidential Transactions. https://elementsproject.org/features/confidential-transactions, 2024.

[15] Jesper Buus Nielsen, Peter Sebastian Nordholt, Claudio Orlandi, and Sai Sheshank Burra. A new approach to practical active-secure two-party computation. In *CRYPTO*, pages 681–700, 2012.

[16] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, pages 129–140, 1991.

[17] Nan Wang, Sid Chi-Kin Chau, and Dongxi Liu. Swiftrange: A short and efficient zero-knowledge range argument for confidential transactions and more. In *SP*, pages 1832–1848, 2024.

[18] Chenkai Weng, Kang Yang, Jonathan Katz, and Xiao Wang. Wolverine: Fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. In *SP*, pages 1074–1091, 2021.

[19] Kang Yang, Pratik Sarkar, Chenkai Weng, and Xiao Wang. Quicksilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In *CCS*, pages 2986–3001, 2021.