

Bulletproofs: The State-of-the-art Range Proofs for Confidential Transactions

EE451: Supervised Research Exposition Project

Suyash Bagad

Guide: Prof. Saravanan Vijayakumaran

Department of Electrical Engineering,
Indian Institute of Technology, Bombay

2019

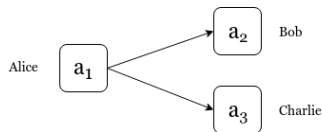
Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

Range Proofs

- ▶ *Range proofs* are proofs that a secret value, which has either been encrypted or committed, lies in a particular interval.
- ▶ They do not leak any other information about the secret value other than that the value lies in a particular range and thus are also referred as **Zero Knowledge Range Proofs**.
- ▶ Real-life examples:
 - ▶ Electronic vote
 - ▶ n candidates for a vote
 - ▶ each voter needs to prove that her private choice belongs to the interval $[1, n]$
 - ▶ Proof that a person's salary is sufficient to rent a house, or obtain a mortgage, without revealing the exact number
 - ▶ Proof that a person is in a country, without revealing his exact location

Why range proofs?



- ▶ Let's say in a transaction where Alice sends some amount of money to Bob and Charlie simultaneously.
- ▶ Alice has a_1 and sends a_2, a_3 to Bob and Charlie respectively. We should thus have $a_1 = a_2 + a_3$.
- ▶ The above fact is verified by an entity called *miners*.
- ▶ Now, let's say Alice attempts a fraudulent transaction by setting $a_1 = 5, a_2 = 6, a_3 = -1$. Here, $a_1 = a_2 + a_3$ is verified, but there has been a generation of new currency!
- ▶ To avoid this, along with the verification, the amounts also need to be in a specified range. Hence, Range Proofs!

Existing Range Proofs

- ▶ The standard method of constructing a range proof [5].
- ▶ For an output amount b , we'll prove $b_i \in \{0, 1\} \forall i \in \{0, \dots, r\}$,

$$b = b_0 2^0 + b_1 2^1 + b_2 2^2 + \dots + b_r 2^r$$

- ▶ To do this, we write the commitments with a masking secret key a , $C = aG + bH$, pick a_0, a_1, \dots, a_r with $\sum_{i=0}^r a_i = a$

$$C_i = a_i G + b_i 2^i H$$

and prove for each i that C_i is a commitment to either 0 or 1, without revealing explicitly which one it is.

- ▶ This is easily done by providing a ring signature on the ring

$$(C_i, C_i - 2^i H)$$

- ▶ If $b_i \notin \{0, 1\}$, then neither of the keys in the above ring will be a commitment to zero, and the ring will not be signable.
- ▶ The prover provides all C_i and the verifier checks that $\sum_i C_i = C$.

Existing Range Proofs

- ▶ Lipmaa [1], [9] proposes a range proof which uses integer commitments and Lagrange's four-square theorem

Lemma (Lagrange's theorem)

Let $X \in \mathbb{Z}$. Then $X \geq 0$ iff $\exists (x_1, x_2, x_3, x_4) \in \mathbb{Z}^4 : X = x_1^2 + x_2^2 + x_3^2 + x_4^2$

- ▶ Groth [2] gave an optimized argument for integers of the form $4y + 1, y \in \mathbb{Z}$, since integers of this form only require 3 squares.

Lemma (Groth's characterisation)

Let X be an integer which is not of the form $4^n(8k + 7)$, where $n, k \in \mathbb{N}$. Then $X \geq 0$ iff $\exists (x_1, x_2, x_3) \in \mathbb{Z}^3 : X = x_1^2 + x_2^2 + x_3^2$

- ▶ These arguments require only a constant number of commitments. However, each commitment is **large**, as the security of the argument relies on the **Strong RSA assumption**. Also, these range proofs require a **trusted setup**.

Blockchain

- Blockchain technology offers a way for untrusted parties to reach a consensus on a common digital history.

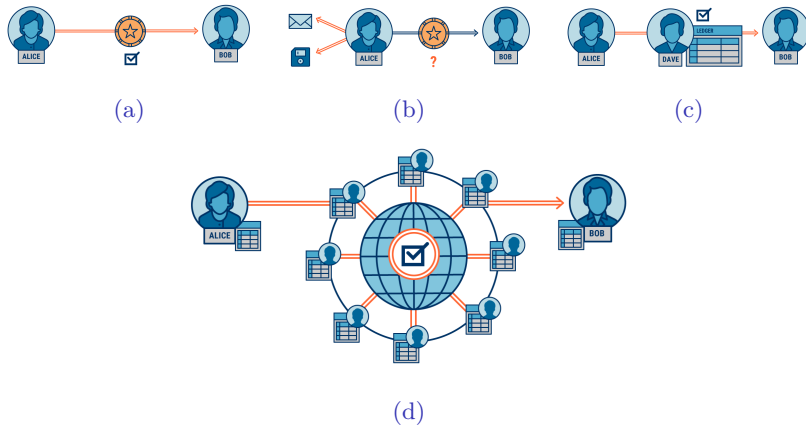


Figure: Sub-figures (a) a physical transaction, (b) a digital transaction, (c) a digital transaction with ledger and (d) a decentralized ledger.

Privacy in Blockchain transactions

- ▶ The privacy for payments needs to have two properties:
 - ▶ *Anonymity*: Hiding the identity of sender and receiver in a transaction
 - ▶ *Confidentiality*: Hiding the amounts transferred
- ▶ In case of Bitcoin, any independent observer can verify the state of the blockchain as well as the validity of all the transactions on the ledger.
- ▶ This is a *serious* limitation for Bitcoin and could possibly prohibit many use cases.
- ▶ As an example, if employees of a company were to receive their salaries in bitcoin, would they agree if their salaries were published on the public blockchain?

Bulletproofs

- ▶ *Short* like a bullet with bulletproof *security* assumptions
- ▶ A zero-knowledge argument of knowledge system, to prove that a secret committed value lies in a given interval.
- ▶ Do not require a *trusted setup* [3], [8].
- ▶ Relies only on the *discrete logarithm assumption*
- ▶ Made non-interactive using the Fiat-Shamir heuristic.
- ▶ Supports a simple and efficient multi-party computation (*MPC*) protocol
- ▶ Useful in *Confidential Transactions, Mimblewimble, Provisions protocol, verifiable shuffle, smart contracts and arithmetic circuit satisfiability*.

Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

Preliminaries I

Definition (Discrete Log Relation)

For all PPT adversaries \mathcal{A} and for all $n \geq 2$, \exists a negligible function $\mu(\lambda)$ s.t

$$\mathbb{P}\left[\begin{array}{l} G = \text{Setup}(1^\lambda), g_1, \dots, g_n \leftarrow \mathbb{G}; \\ a_1, \dots, a_n \in \mathbb{Z}_p \leftarrow \mathcal{A}(\mathbb{G}, g_1, \dots, g_n) \end{array} : \exists a_i \neq 0 \wedge \prod_{i=1}^n g_i^{a_i} = 1 \right] \leq \mu(\lambda)$$

We say $\prod_{i=1}^n g_i^{a_i} = 1$ is a non trivial discrete log relation between g_1, \dots, g_n .

Definition (Commitments)

A non-interactive commitment consists of two PPT algorithms $(\text{Setup}, \text{Com})$. For a message $x \in M_{pp}$ (message space), the algorithm proceeds as follows:

- ▶ public parameters $pp \leftarrow \text{Setup}(1^\lambda)$ for security parameter λ
- ▶ $\text{Com}_{pp} : M_{pp} \times R_{pp} \rightarrow C_{pp}$, where R_{pp} is randomness space
- ▶ $r \leftarrow R_{pp}$ and compute **com** = $\text{Com}_{pp}(x; r)$

Preliminaries II

Definition (Homomorphic Commitments)

A homomorphic commitment is a non-interactive commitment such that M_{pp} , R_{pp} , C_{pp} are all abelian groups, and $\forall x_1, x_2 \in M_{pp}, r_1, r_2 \in R_{pp}$, we have

$$\text{Com}(x_1; r_1) + \text{Com}(x_2; r_2) = \text{Com}(x_1 + x_2; r_1 + r_2)$$

Definition (Hiding Commitment)

A commitment scheme is said to be hiding if for all PPT adversaries \mathcal{A} , $\exists \mu(\lambda)$, a negligible function such that,

$$\left| \mathbb{P} \left[b' = b \mid \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda); \\ (x_0, x_1) \in M_{pp}^2 \leftarrow \mathcal{A}(pp), b \leftarrow \{0, 1\}, r \leftarrow R_{pp}, \\ \text{com} = \text{Com}(x_b; r), b' \leftarrow \mathcal{A}(pp, \text{com}) \end{array} \right] - \frac{1}{2} \right| \leq \mu(\lambda)$$

where the probability is over b', r, Setup and \mathcal{A} . For perfectly hiding schemes, $\mu(\lambda) = 0$.

Preliminaries III

Definition (Binding Commitment)

A commitment scheme is said to be binding if for all PPT adversaries \mathcal{A} , $\exists \mu(\lambda)$, a negligible function such that,

$$\mathbb{P} \left[\text{Com}(x_0; r_0) = \text{Com}(x_1; r_1) \wedge x_0 \neq x_1 \mid \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda), \\ x_0, x_1, r_0, r_1 \leftarrow \mathcal{A}(pp) \end{array} \right] \leq \mu(\lambda)$$

where the probability is over Setup and \mathcal{A} . Again, if $\mu(\lambda) = 0$ then we say the scheme is perfectly binding.

Definition (Pedersen Commitment)

$M_{pp}, R_{pp} = \mathbb{Z}_p$, $C_{pp} = \mathbb{G}$ of order p .

- ▶ Setup: $g, h \leftarrow \mathbb{G}$
- ▶ $\text{Com}(x; r) = (g^x h^r)$

Preliminaries IV

Definition (Pedersen Vector Commitment)

$M_{pp} = \mathbb{Z}_p^n, R_{pp} = \mathbb{Z}_p, C_{pp} = \mathbb{G}$ of order p .

- ▶ *Setup:* $\mathbf{g} = (g_1, \dots, g_n), h \leftarrow \mathbb{G}$
- ▶ $Com(\mathbf{x} = (x_1, \dots, x_n); r) = (h^r \mathbf{g}^{\mathbf{x}})$

Note: The Pedersen vector commitment is **perfectly hiding** and **computationally binding** under the discrete logarithm assumption.

Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

Zero-Knowledge Arguments

- ▶ A protocol in which a *prover* convinces a *verifier* that a statement is true *without* revealing any information about why it holds
- ▶ An argument is a proof only if the prover is computationally bounded and some computational hardness holds.
- ▶ Three interacting PPT algorithms, namely $(\text{Setup}, \mathcal{P}, \mathcal{V})$,
 - ▶ Setup: $\sigma \leftarrow \text{Setup}(1^\lambda)$, σ is common reference string
 - ▶ \mathcal{P} : prover, \mathcal{V} : verifier
 - ▶ Transcript $tr \leftarrow \langle \mathcal{P}, \mathcal{V} \rangle$
 - ▶ $\langle \mathcal{P}, \mathcal{V} \rangle = b$, $b = 0$ if the verifier rejects or $b = 1$ accepts
- ▶ Define language $\mathcal{L}_\sigma = \{x \mid \exists w : (\sigma, x, w) \in \mathcal{R}\}$ as the set of statements x those have a witness w in a polynomial-time ternary relation $\mathcal{R} \subseteq \{0, 1\}^* \times \{0, 1\}^* \times \{0, 1\}^*$

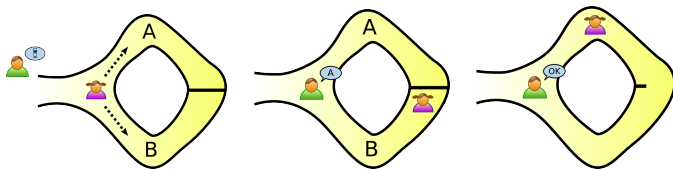


Figure: Image courtesy: <https://en.wikipedia.org/wiki/Zero-knowledgeproof>

Defining Arguments of Knowledge

Definition (Argument of Knowledge)

*The triple $(\text{Setup}, \mathcal{P}, \mathcal{V})$ is called an argument of knowledge for relation \mathcal{R} if it is **perfectly complete** and has **computational witness-extended emulation**.*

Definition (Perfect completeness)

$(\text{Setup}, \mathcal{P}, \mathcal{V})$ has perfect completeness if for all non-uniform polynomial time adversaries \mathcal{A}

$$\mathbb{P}\left[(\sigma, u, w) \notin \mathcal{R} \text{ or } \langle \mathcal{P}(\sigma, u, w), \mathcal{V}(\sigma, u) \rangle = 1 \mid \begin{array}{l} \sigma \leftarrow \text{Setup}(1^\lambda) \\ (u, w) \leftarrow \mathcal{A}(\sigma) \end{array} \right] = 1$$

- *Perfect completeness* implies that if a statement is actually true, then an honest verifier is convinced with probability 1 about the truth of the statement by an honest prover.

Defining Arguments of Knowledge

Definition (Computational Witness-Extended Emulation)

(Setup, P, V) has witness-extended emulation if for all deterministic polynomial time P^ there exists an expected polynomial time emulator \mathcal{E} such that for all pairs of interactive adversaries $\mathcal{A}_1, \mathcal{A}_2$ there exists a negligible function $\mu(\lambda)$ such that*

$$\left| \mathbb{P} \left[A_1(tr) = 1 \mid \begin{array}{l} \sigma \leftarrow \text{Setup}(1^\lambda,) \\ (u, s) \leftarrow \mathcal{A}_2(\sigma), \\ tr \leftarrow \langle (\mathcal{P}^*(\sigma, u, s), V(u, s)) \rangle \end{array} \right] - \right. \\ \left. \mathbb{P} \left[\begin{array}{l} A_1(tr) = 1 \wedge \\ (tr \text{ accepted} \implies (\sigma, u, w) \in \mathcal{R}) \end{array} \mid \begin{array}{l} \sigma \leftarrow \text{Setup}(1^\lambda,) \\ (u, s) \leftarrow \mathcal{A}_2(\sigma), \\ (tr, w) \leftarrow \mathcal{E}^{\mathcal{O}}(\sigma, u) \end{array} \right] \right| \leq \mu(\lambda)$$

where the oracle is given by $\mathcal{O} = \langle (\mathcal{P}^(\sigma, u, s), V(u, s)) \rangle$, and permits rewinding to a specific point and resuming with fresh randomness for the verifier from this point onwards. We can also define computational witness-extended emulation by restricting to non-uniform polynomial time adversaries \mathcal{A}_1 and \mathcal{A}_2 .*

More on Arguments of Knowledge

- ▶ Computational witness-extended emulation implies that when an adversary produces an argument to convince the verifier with some probability, then we have a corresponding emulator producing identically distributed argument with same probability, but also a witness.

Definition (Public coin)

An argument of knowledge $(\text{Setup}, \mathcal{P}, \mathcal{V})$ is called public coin if all messages sent from the verifier to the prover are chosen uniformly at random and independent of the prover's messages, i.e., the challenges correspond to the verifier's randomness ρ .

Definition (Zero Knowledge Argument of Knowledge)

An argument of knowledge $(\text{Setup}, \mathcal{P}, \mathcal{V})$ is zero knowledge if it reveals no information about w apart from what could be deduced from the fact that $(\sigma, u, w) \in \mathcal{R}$.

More on Arguments of Knowledge

Definition (Perfect Special Honest-Verifier Zero-Knowledge)

A public coin argument of knowledge $(Setup, \mathcal{P}, \mathcal{V})$ is a perfect special honest verifier zero knowledge (SHVZK) argument of knowledge for \mathcal{R} if there exists a probabilistic polynomial time simulator \mathcal{S} such that for all pairs of interactive adversaries $\mathcal{A}_1, \mathcal{A}_2$

$$\begin{aligned} \mathbb{P} \left[(\sigma, u, w) \in \mathcal{R} \wedge \mathcal{A}_1(tr) = 1 \mid \begin{array}{l} \sigma \leftarrow Setup(1^\lambda,) \\ (u, w, \rho) \leftarrow \mathcal{A}_2(\sigma), \\ tr \leftarrow \langle (\mathcal{P}(\sigma, u, s), V(\sigma, u; \rho)) \rangle \end{array} \right] \\ = \mathbb{P} \left[(\sigma, u, w) \in \mathcal{R} \wedge \mathcal{A}_1(tr) = 1 \mid \begin{array}{l} \sigma \leftarrow Setup(1^\lambda,) \\ (u, w, \rho) \leftarrow \mathcal{A}_2(\sigma), \\ tr \leftarrow \mathcal{S}(u, \rho) \end{array} \right] \end{aligned}$$

- PSHVZK AoK implies that even if an adversary chooses a distribution over statements and witnesses, it isn't able to distinguish between simulated transcript and honestly generated transcript for $u \in \mathcal{L}_\sigma$.

Finally!

Definition (Zero-Knowledge Range Proof)

Given a commitment scheme $(Setup, Com)$ over a message space M_{pp} , which is a set with a total ordering, a Zero-Knowledge Range Proof is a SHVZK argument of knowledge for the relation \mathcal{R}_{Range} :

$$\mathcal{R}_{Range} : (pp, (\mathbf{com}, l, r), (x, \rho)) \in \mathcal{R}_{Range} \leftrightarrow \mathbf{com} = Com(x; \rho) \wedge x \in [l, r]$$

**THAT'S TOO MUCH
OF MATH...**

**I THINK I NOW HAVE
ZERO KNOWLEDGE!**

Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

Inner-Product Argument

- ▶ The inner product argument is an efficient proof system for the following relation:

$$\{(\boxed{\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, P \in \mathbb{G}, c \in \mathbb{Z}_p}; \boxed{\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n}) : \boxed{P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} \wedge c = \langle \mathbf{a}, \mathbf{b} \rangle}\} \quad (1)$$

- ▶ The simplest proof system for (1) is: \mathcal{P} sends $\mathcal{V}(\mathbf{a}, \mathbf{b}) \in \mathbb{Z}_p^n$, requiring to send $2n$ elements to \mathcal{V} .
- ▶ Let's design a proof system for the relation:

$$\{(\boxed{\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, u, P \in \mathbb{G}}; \boxed{\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n}) : \boxed{P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} \cdot u^{\langle \mathbf{a}, \mathbf{b} \rangle}}\} \quad (2)$$

- ▶ Define a hash function $H : \mathbb{Z}_p^{2n+1} \rightarrow \mathbb{G}$, $\mathbf{a}_1, \mathbf{a}'_1, \mathbf{b}_1, \mathbf{b}'_1 \in \mathbb{Z}_p^{n/2}, c \in \mathbb{Z}_p$

$$H(\mathbf{a}_1, \mathbf{a}'_1, \mathbf{b}_1, \mathbf{b}'_1, c) = \mathbf{g}_{[:n/2]}^{\mathbf{a}_1} \cdot \mathbf{g}_{[n/2:]}^{\mathbf{a}'_1} \cdot \mathbf{h}_{[:n/2]}^{\mathbf{b}_1} \cdot \mathbf{h}_{[n/2:]}^{\mathbf{b}'_1} \cdot u^c \in \mathbb{G}$$

- ▶ H is additively homomorphic in its inputs.
- ▶ Let $n' = n/2$, $\mathbf{a} = (\mathbf{a}_{[:n']}, \mathbf{a}_{[n'::]})$, $\mathbf{b} = (\mathbf{b}_{[:n']}, \mathbf{b}_{[n'::]})$
- ▶ Also, note that $P = H(\mathbf{a}_{[:n']}, \mathbf{a}_{[n':]}, \mathbf{b}_{[:n']}, \mathbf{b}_{[n':]}, \langle \mathbf{a}, \mathbf{b} \rangle)$

Algorithm 1: Proof system for relation (2)

Input
 $(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, u, P \in \mathbb{G}; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n)$

$(\mathbf{g}, \mathbf{h}, u, P, \mathbf{a}, \mathbf{b})$



Prover

$(\mathbf{g}, \mathbf{h}, u, P)$



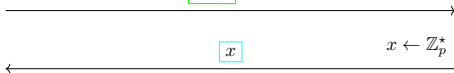
Verifier

$$n' = n/2$$

$$L = H(\mathbf{0}^{n'}, \mathbf{a}_{[n']}, \mathbf{b}_{[n']}, \mathbf{0}^{n'}, \langle \mathbf{a}_{[n']}, \mathbf{b}_{[n']} \rangle)$$

$$R = H(\mathbf{a}_{[n']}, \mathbf{0}^{n'}, \mathbf{0}^{n'}, \mathbf{b}_{[n']}, \langle \mathbf{a}_{[n']}, \mathbf{b}_{[n']} \rangle)$$

$$L, R \in \mathbb{G}$$



$$\mathbf{a}' = x \cdot \mathbf{a}_{[n']} + x^{-1} \cdot \mathbf{a}_{[n']}$$

$$\mathbf{b}' = x^{-1} \cdot \mathbf{b}_{[n']} + x \cdot \mathbf{b}_{[n']}$$

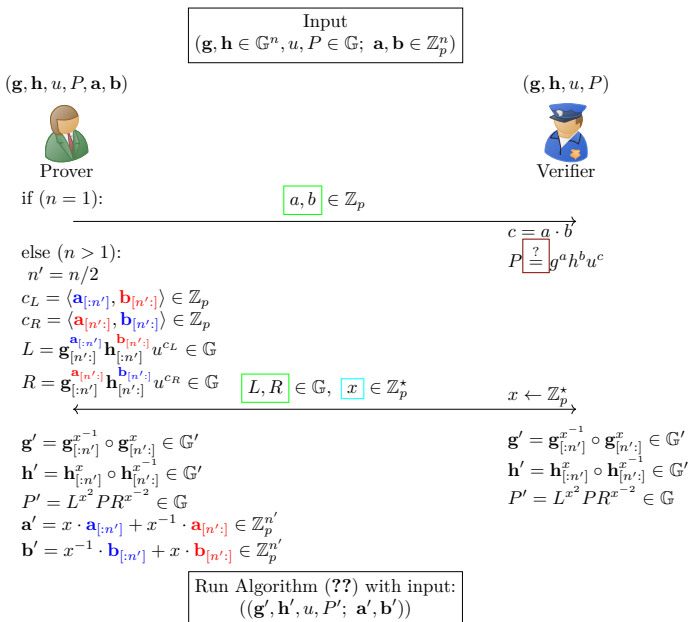
$$\mathbf{a}', \mathbf{b}' \in \mathbb{Z}_p^{n'}$$



$$P' = L^{(x^2)} \cdot P \cdot R^{(x^{-2})}$$

$$P' \stackrel{?}{=} H(x^{-1}\mathbf{a}', x\mathbf{a}', x\mathbf{b}', x^{-1}\mathbf{b}', \langle \mathbf{a}', \mathbf{b}' \rangle)$$

Algorithm 2: Proof system for relation (2)



Inner-Product Argument

Algorithm 4: Proof system for Relation (1)

Input: $(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, P \in \mathbb{G}, c \in \mathbb{Z}_p; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n)$

1: \mathcal{P} 's inputs: $(\mathbf{g}, \mathbf{h}, P, c, \mathbf{a}, \mathbf{b})$

2: \mathcal{V} 's inputs: $(\mathbf{g}, \mathbf{h}, P, c,)$

Output: $\{\mathcal{V} \text{ accepts or } \mathcal{V} \text{ rejects}\}$

Protocol:

3: \mathcal{V} computes :

4: $x \leftarrow \mathbb{Z}_p^*$

5: $\mathcal{V} \rightarrow \mathcal{P} : x$

6: $P' = P \cdot u^{x \cdot c}$

7: Run Algorithm 3 with **Input:** $(\mathbf{g}, \mathbf{h}, u^x, P'; \mathbf{a}, \mathbf{b}) = 0$

Theorem (Inner-Product Argument)

The argument presented in Algorithm 4 for the relation (1) has perfect completeness and statistical witness-extended-emulation for either extracting a non-trivial discrete logarithm relation between $\mathbf{g}, \mathbf{h}, u$ or extracting a valid witness \mathbf{a}, \mathbf{b} .

Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

Inner-Product Range Proof

- ▶ We wish to design a proof system for the following relation which is equivalent to the range proof relation

$$\{(g, h \in \mathbb{G}, V, n; v, \gamma \in \mathbb{Z}_p) : V = g^v h^\gamma \wedge v \in [0, 2^n - 1]\} \quad (3)$$

- ▶ Let $\mathbf{a}_L = (a_1, \dots, a_n) \in \{0, 1\}^n$ be the vector containing the bits of v , $v = \langle \mathbf{a}_L, \mathbf{2}^n \rangle$. Prover \mathcal{P} convinces the verifier that $v \in [0, 2^n - 1]$ by proving that:
 - ▶ It knows $\mathbf{a}_L \in \mathbb{Z}_p^n$, $v, \gamma \in \mathbb{Z}_p$ such that $V = g^v h^\gamma$
 - ▶ $\langle \mathbf{a}_L, \mathbf{2}^n \rangle = v$, $\wedge \mathbf{a}_L \circ \mathbf{a}_R = \mathbf{0}^n \wedge \mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^n$
- ▶ To do so, we take a random linear combination (chosen by the verifier) of the constraints to use inner-product argument.
- ▶ Note that $\langle \mathbf{b}, \mathbf{y}^n \rangle = 0, y \in \mathbb{Z}_p \implies \mathbf{b} = \mathbf{0}^n$
- ▶ For randomly chosen $y, z \in \mathbb{Z}_p$, we can write:

$$z^2 \cdot \langle \mathbf{a}_L, \mathbf{2}^n \rangle + z \cdot \langle \mathbf{a}_L - \mathbf{1}^n - \mathbf{a}_R, \mathbf{y}^n \rangle + \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n \rangle = z^2 \cdot v \quad (4)$$

$$\implies \left\langle \boxed{\mathbf{a}_L - z \cdot \mathbf{1}^n}, \boxed{\mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n) + z^2 \cdot \mathbf{2}^n} \right\rangle = z^2 + \delta(y, z) \quad (5)$$

$$\delta(y, z) = (z - z^2) \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle - z^3 \langle \mathbf{1}^n, \mathbf{2}^n \rangle$$

Inner-Product Range Proof

- Define two linear vector polynomials $l(X), r(X) \in \mathbb{Z}_p^n[X]$ and $t(X) \in \mathbb{Z}_p$ as follows:

$$l(X) = \mathbf{a}_L - z \cdot \mathbf{1}^n + s_L \cdot X$$

$$r(X) = \mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n) + s_R \cdot X + z^2 \cdot \mathbf{2}^n$$

$$t(X) = \langle l(X), r(X) \rangle = t_0 + t_1 \cdot X + t_2 \cdot X^2$$

- The prover needs to convince the verifier that this t_0 satisfies

$$t_0 = z^2 \cdot v + \delta(y, z)$$

Algorithm 6: Inner-Product Range Proof

Input
 $((\mathbf{g}, \mathbf{h} \in \mathbb{G}, V, n; v, \gamma \in \mathbb{Z}_p))$

$(\mathbf{g}, \mathbf{h}, V, n, v, \gamma)$



Prover

$(\mathbf{g}, \mathbf{h}, V, n)$



Verifier

$$\mathbf{a}_L \in \{0, 1\}^n \text{ s.t. } \langle \mathbf{a}_L, \mathbf{2}^n \rangle = v$$

$$\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^n \in \mathbb{Z}_p^n$$

$$\alpha \leftarrow \mathbb{Z}_p$$

$$A = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R}$$

$$\mathbf{s}_L, \mathbf{s}_R \leftarrow \mathbb{Z}_p^n$$

$$\rho \leftarrow \mathbb{Z}_p$$

$$S = h^\rho \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}$$

$A, S \in \mathbb{G}$

→

y, z

$y, z \leftarrow \mathbb{Z}_p^*$

←

$$\tau_1, \tau_2 \leftarrow \mathbb{Z}_p$$

$$T_i = g^{t_i} h^{\tau_i}, i \in \{1, 2\}$$

$T_1, T_2 \in \mathbb{G}$

→

x

$x \leftarrow \mathbb{Z}_p^*$

←

Algorithm 5: Inner-Product Range Proof (Continued)



Prover



Verifier

$$\begin{aligned}\mathbf{l} &= l(x) = \mathbf{a}_L - z \cdot \mathbf{1}^n + \mathbf{s}_L \cdot x \in \mathbb{Z}_p^n \\ \mathbf{r} &= r(x) = \mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n + \mathbf{s}_R \cdot X) + z^2 \cdot \mathbf{2}^n \in \mathbb{Z}_p^n \\ \hat{t} &= \langle \mathbf{l}, \mathbf{r} \rangle \in \mathbb{Z}_p \\ \tau_x &= \tau_2 \cdot x^2 + \tau_1 \cdot x + z^2 \cdot \gamma \in \mathbb{Z}_p \\ \mu &= \alpha + \rho \cdot x \in \mathbb{Z}_p\end{aligned}$$

$$\tau_x, \mu, \hat{t} \in_p, \mathbf{l}, \mathbf{r} \in_p^n$$

$$h'_i = h_i^{(y^{-i+1})} \in \mathbb{G}, \forall i$$

$$g^{\hat{t} h^{\tau_x}} = V^{z^2} \cdot g^{\delta(y,z)} \cdot T_1^x \cdot T_2^{x^2}$$

$$P = A \cdot S^x \cdot \mathbf{g}^{-z} \cdot (\mathbf{h}')^z \cdot \mathbf{y}^n + z^2 \cdot \mathbf{2}^n \in \mathbb{G}$$

$$P = h^\mu \cdot \mathbf{g}^1 \cdot (\mathbf{h}')^{\mathbf{r}}$$

$$\hat{t} = \langle \mathbf{l}, \mathbf{r} \rangle \in \mathbb{Z}_p$$

Inner-Product Range Proof

- ▶ In this protocol, \mathcal{P} transmits (\mathbf{l}, \mathbf{r}) , whose size is linear in n .
- ▶ The total communication cost in this protocol is $n + 2$ elements in \mathbb{G} and 3 elements in \mathbb{Z}_p .
- ▶ It remains for the verifier to check if the received \mathbf{l}, \mathbf{r} are actually $l(x), r(x)$ and also check if $t(x) = \langle \mathbf{l}, \mathbf{r} \rangle$.
- ▶ To do so, first, \mathcal{V} switches the generators of the commitment from $\mathbf{h} \in \mathbb{G}^n$ to $\mathbf{h}' = \mathbf{h}(\mathbf{y}^{-n})$.
- ▶ Now, A becomes a commitment to $(\mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n)$ w.r.t $(\mathbf{g}, \mathbf{h}', h)$. Similarly, S is now a vector commitment to $(\mathbf{s}_L, \mathbf{s}_R \circ \mathbf{y}^n)$.

Logarithmic Range Proof

- ▶ For a proof whose size is logarithmic in n , we can eliminate the transfer of \mathbf{l} and \mathbf{r} using the inner-product argument.
- ▶ Vectors (\mathbf{l}, \mathbf{r}) are not secret and hence a protocol that only provides soundness is sufficient.
- ▶ We observe that verifying lines (5),(6) of Part III of Algorithm 4 is the same as verifying that the witness (\mathbf{l}, \mathbf{r}) satisfies the inner product relation (1) on public input $(\mathbf{g}, \mathbf{h}', Ph^{-\mu}, \hat{t})$, where P commitment to two vectors $\mathbf{l}, \mathbf{r} \in \mathbb{Z}_p^n$ whose inner product is \hat{t} .

$$P \stackrel{?}{=} h^\mu \cdot \mathbf{g}^{\mathbf{l}} \cdot (\mathbf{h}')^{\mathbf{r}}$$

$$\hat{t} \stackrel{?}{=} \langle \mathbf{l}, \mathbf{r} \rangle \in \mathbb{Z}_p$$

$$\{(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, P \in \mathbb{G}, c \in \mathbb{Z}_p; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n) : P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} \wedge c = \langle \mathbf{a}, \mathbf{b} \rangle\}$$

- ▶ Thus, using the inner product argument, the total communication cost reduces down to $2\lceil \log_2(n) \rceil + 2 + 2$ elements in \mathbb{G} and 5 elements in \mathbb{Z}_p .

Aggregating Logarithmic Proofs

- ▶ In many of the range proof applications described in, a single prover needs to perform multiple range proofs at the same time.

$$\{(g, h \in \mathbb{G}, V \in \mathbb{G}^m; \mathbf{v}, \boldsymbol{\gamma} \in \mathbb{Z}_p^m) : V_j = g^{v_j} h^{\gamma_j} \wedge v_j \in [0, 2^n - 1] \forall j \in [1, m]\}$$

- ▶ Now, \mathbf{a}_L is the concatenation of all of the bits for every v_j . Thus,

$$l(X) = (\mathbf{a}_L - z \cdot \mathbf{1}^{nm}) + \mathbf{s}_L \cdot X \in \mathbb{Z}_p^{nm}[X]$$

$$r(X) = \mathbf{y}^{nm} \circ (\mathbf{a}_R + z \cdot \mathbf{1}^{nm} + \mathbf{s}_R \cdot X) + \sum_{j=1}^m z^{1+j} \cdot (\mathbf{0}^{(j-1)n} \parallel \mathbf{2}^n \parallel \mathbf{0}^{(m-j)n})$$

$$\tau_x = \tau_1 \cdot x + \tau_2 \cdot x^2 + \sum_{j=1}^m z^{1+j} \cdot \gamma_j$$

- ▶ This aggregated range proof uses $2\lceil \log_2(nm) \rceil + 4$ elements in \mathbb{G} as opposed to $m(2\lceil \log_2(n) \rceil + 4)$ for m independent range proofs.

Theorem (Aggregate Range Proof)

The aggregate range proof presented above has perfect completeness, perfect honest verifier zero-knowledge and computational witness extended emulation.

Fiat-Shamir Heuristic for Non-Interactive Proof

- ▶ The verifier is a public coin verifier, as all the honest verifier's messages are random elements from \mathbb{Z}_p^* .
- ▶ We can therefore convert the protocol into a non-interactive protocol that is secure and full zero-knowledge in the random oracle model using the Fiat-Shamir heuristic [4].
- ▶ To do so, we just need to replace all random challenges by hashes of the transcript up to that point. For example, $y = H(A, S)$ and $z = H(A, S, y)$.
- ▶ For a **non-trusted setup** we can use such a hash function to generate the public parameters.

MPC Protocol for Bulletproofs

- ▶ In several of the applications, the prover could potentially consist of multiple parties (say, m) who each want to generate a single range proof. (Note the difference from aggregating range proofs)
- ▶ The MPC protocol works as follows:
 - ▶ Assign a set of distinct generators $(\mathbf{g}^{(k)}, \mathbf{h}^{(k)})_{k=1}^m$ to each party
 - ▶ Define $\mathbf{g} = (\mathbf{g}^{(1)} \parallel \mathbf{g}^{(2)} \parallel \dots \parallel \mathbf{g}^{(m)})$, $g_i = g_{\lceil i/m \rceil}^{((i-1) \bmod m + 1)}$
 - ▶ Define $\mathbf{h}, \mathbf{h}^{(k)}$ similarly

- ▶ As described in algorithm 5, in each of its three rounds, each party generates its part of the proof, i.e.

$$A^{(k)}, S^{(k)}; T_1^{(k)}, T_2^{(k)}; \tau_x^{(k)}, \mu^{(k)}, \hat{t}^{(k)}, \mathbf{l}^{(k)}, \mathbf{r}^{(k)}$$

- ▶ These shares are then sent to a dealer who simply adds them homomorphically to generate the respective proof component, i.e. $A = \prod_{k=1}^m A^{(k)}$ and $\tau_x = \sum_{k=1}^m \tau_x^{(k)}$.
- ▶ Then, each party sends $\mathbf{l}^{(k)}, \mathbf{r}^{(k)}$ to the dealer who computes \mathbf{l}, \mathbf{r} as the interleaved concatenation of the shares.
- ▶ Finally, the inner product argument is run and the final proof is generated.

Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

Theoretical Performance I

- Summary of proof sizes for different protocols we used to build Bulletproofs

	# \mathbb{G} elements	# \mathbb{Z}_p elements
Simplest IPA	$2n$	0
IPA I	n	2
IIPA	$2(\lceil \log_2(n) \rceil)$	2
IPRP using IPA	$2n + 4$	3
IPRP using IIPA	$2(\lceil \log_2(n) \rceil) + 4$	5
ARP	$m \times \{2(\lceil \log_2(n) \rceil) + 4\}$	$5m$
Improved ARP	$2\{\lceil \log_2(n) \rceil + \lceil \log_2(m) \rceil\} + 4$	5

Table: Proof sizes for different inner product and ARP protocols.

Theoretical Performance II

	# \mathbb{G} elements	# \mathbb{Z}_p elements
Σ Protocol [6]	mn	$3mn + 1$
Poelstra [7]	$0.63mn$	$1.26mn + 1$
Bulletproofs	$2(\log_2(n) + \log_2(m)) + 4$	5






Table: Range proof size for m proofs and range $[0, 2^n - 1]$.

- ▶ Bulletproofs have a significant advantage when providing multiple range proofs at once.
- ▶ The proof size for the protocol (5) only grows by an additive logarithmic factor when conducting m range proofs, while all other solutions grow multiplicatively in m .






Table of Contents

- ① Introduction
 - Range Proofs
 - Bulletproofs
- ② Preliminaries
- ③ Zero-Knowledge Arguments
- ④ Improved Inner-Product Argument
- ⑤ Range Proof Protocol with Logarithmic Size
 - Inner-Product Range Proof
 - Logarithmic Range Proof
 - Aggregating Logarithmic Proofs
 - Fiat-Shamir Heuristic for Non-Interactive Proof
 - MPC Protocol for Bulletproofs
- ⑥ Performance
- ⑦ References

References I

-  [Helger Lipmaa](#). On diophantine complexity and statistical zero-knowledge arguments. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 398–415. Springer, 2003.
-  [Jens Groth](#). Non-interactive zero-knowledge arguments for voting. In *International Conference on Applied Cryptography and Network Security*, pages 467–482. Springer, 2005.
-  [Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.](#): Zerocash: Decentralized anonymous payments from bitcoin. In *IACR Cryptology ePrint Archive* 2014, 349 (2014)
-  [Mihir Bellare and Phillip Rogaway](#). Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93*, pages 62–73, 1993.
-  [Maxwell Gregory](#). Confidential Transactions. (2015), https://people.xiph.org/~greg/confidential_values.txt

References II

-  [Ronald Cramer and Ivan Damgard](#). Zero-knowledge proofs for finite field arithmetic, or: Can zero-knowledge be for free?. In *CRYPTO 98*, pages 424–441. Springer, 1998.
-  [Andrew Poelstra, Adam Back, Mark Friedenbach, Gregory Maxwell, and Pieter Wuille](#). Confidential assets.
-  [Mine Z-cash](#)
<https://minezcash.com/zcash-trusted-setup/>
-  [Canard S., Traore J., Jambert A., Coisel I.](#) On the Practical Use of Range Proofs, In *Provable Privacy Workshop*, July, 2012.
-  [CB Insights](#)
<https://www.cbinsights.com/research/what-is-blockchain-technology/>