Bulletproofs: Efficient Range Proofs for Confidential Transactions

Benedikt Bünz*[1], Jonathan Bootle†[2], Dan Boneh‡[1], Andrew Poelstra§[3], Pieter Wuille¶[3], and Greg Maxwell‖[3]

[1]Stanford University
[2]University College London
[3]Blockstream

Abstract

We propose Bulletproofs, a new non-interactive zero-knowledge proof protocol with very short proofs and without a trusted setup; the proof size is only logarithmic in the witness size. Bulletproofs are especially well suited for efficient range proofs on committed values: they enable proving that a committed value is in a range using only $2\log_2(n) + 9$ group and field elements, where n is the bit length of the range. Proof generation and verification times are linear in n.

Bulletproofs greatly improve on the linear (in n) sized range proofs currently used to implement Confidential Transactions (CT) in Bitcoin and other cryptocurrencies. Moreover, Bulletproofs supports aggregation of range proofs, so that a party can prove that m commitments lie within a given range by providing only an additive $O(\log(m))$ group elements over the length of a *single* proof. To aggregate proofs from multiple parties, we enable the parties to generate a single proof without revealing their inputs to each other via a simple multi-party computation (MPC) protocol for constructing Bulletproofs. This MPC protocol uses either a constant number of rounds and linear communication, or a logarithmic number of rounds and logarithmic communication.

Bulletproofs build on the techniques of Bootle et al. (EUROCRYPT 2016). Beyond range proofs, Bulletproofs provide short zero-knowledge proofs for general arithmetic circuits while only relying on the discrete logarithm assumption and without requiring a trusted setup. We discuss many applications that would benefit from Bulletproofs, primarily in the

*buenz@cs.stanford.edu
†jonathan.bootle.14@ucl.ac.uk
‡dabo@cs.stanford.edu
§apoelstra@blockstream.io
¶pieter@blockstream.com
‖greg@xiph.org

area of cryptocurrencies. The efficiency of Bulletproofs is particularly well suited for the distributed and trustless nature of blockchains.

1 Introduction

Blockchain-based cryptocurrencies enable peer-to-peer electronic transfer of value by maintaining a global distributed but synchronized ledger, the *blockchain*. Any independent observer can verify both the current state of the blockchain as well as the validity of all transactions on the ledger. In Bitcoin, this innovation requires that all details of a transaction are public: the sender, the receiver, and the amount transferred. In general, we separate privacy for payments into two properties: (1) anonymity, hiding the identities of sender and receiver in a transaction and (2) confidentiality, hiding the amount transferred. While Bitcoin provides some weak anonymity through the unlinkability of Bitcoin addresses to real world identities, it lacks any confidentiality. This is a serious limitation for Bitcoin and could be prohibitive for many use cases. Would employees want to receive their salaries in bitcoin if it meant that their salaries were publicly visible?

To address the confidentiality of transaction amounts, Maxwell [Max16] introduced *confidential transactions* (CT), in which every transaction amount involved is hidden from public view using a commitment to the amount. This approach seems to prevent public validation of the blockchain; an observer can no longer check that the sum of transaction inputs is greater than the sum of transaction outputs, and that all transaction values are positive. This can be addressed by including a zero-knowledge proof of the validity of each confidential transaction.

Current proposals for CT zero-knowledge proofs [PBF+] have either been prohibitively large or required a trusted setup. Neither is desirable. While one could use succinct zero-knowledge proofs (SNARKs) [BSCG+13], they all require a trusted setup, which means that everyone needs to trust that the setup was performed correctly.

Short non-interactive zero-knowledge proofs without a trusted setup, as described in this paper, have many applications in the realm of cryptocurrencies. In any distributed system where proofs are transmitted over a network or stored for a long time, short proofs reduce overall cost.

1.1 Our Contributions

We present Bulletproofs, a new zero-knowledge argument of knowledge[1] system, to prove that a secret committed value lies in a given interval. Bulletproofs do not require a trusted setup. They rely only on the discrete logarithm assumption and are made non-interactive using the Fiat-Shamir heuristic.

[1] Proof systems with computational soundness like Bulletproofs are sometimes called argument systems. We will use the terms proof and argument interchangeably.

Bulletproofs builds on the techniques of Bootle et al. [BCC+16], which yield communication-efficient zero-knowledge proofs. We present a replacement for their inner-product argument that reduces overall communication by a factor of 3. We make Bulletproofs suitable for proving statements on committed values. Examples include a range proof, a verifiable shuffle, and other applications discussed below. We note that a range proof using the protocol of [BCC+16] would have required implementing the commitment opening algorithm as part of the verification circuit.

Further, we give a simple and efficient multi-party computation (MPC) protocol that allows multiple parties with secret committed values to generate a single small range proof for all their values. One version of our MPC protocol is constant-round but with linear communication. Another variant requires only logarithmic communication, but uses a logarithmic number of rounds. This MPC protocol can be used to aggregate the proofs needed for a confidential transaction constructed by multiple parties, into a single short proof.

While we focus on confidential transactions, where our work translates to significant practical savings, we stress that the improvements are not limited to CT. We present Bulletproofs for general NP languages. The proof size is *logarithmic* in the number of multiplication gates in the arithmetic circuit for verifying a witness. The proofs are much shorter than [BCC+16] and allow inputs to be Pedersen commitments to elements of the witness.

Finally, we provide a complete implementation of Bulletproofs. In Section 6 we provide efficiency comparisons with the range proofs currently used for confidential transactions [Max16, Poe] and with other systems. Our implementation includes a general tool for constructing Bulletproofs for any NP language. The tool reads in arithmetic circuits in the Pinocchio [PHGR13] format which lets users use their toolchain. This toolchain includes a compiler from C to the circuit format. We expect this to be of great use to implementers who want to use Bulletproofs.

1.2 Applications

We first discuss several applications for Bulletproofs along with related work specific to these applications. Additional related work is discussed in Section 1.3.

1.2.1 Confidential Transactions and Mimblewimble

Bitcoin and other similar cryptocurrencies use a transaction-output-based system where each transaction fully spends the outputs of one or more previously unspent transactions. These unspent transaction outputs are called UTXOs. Bitcoin allows a single UTXO to be spent to many distinct outputs, each associated with a different address. To spend a UTXO a user must provide a signature, or more precisely a *scriptSig*, that enables the transaction SCRIPT to evaluate to true [BMC+15]. Apart from the validity of the *scriptSig*, miners verify that the transaction spends previously unspent outputs, and that the sum of the inputs is greater than the sum of the outputs.

Maxwell [Max16] introduced the notion of a *confidential transaction*, where the input and output amounts in a transaction are hidden in a Pedersen commitment [P+91]. To enable public validation, the transaction contains a zero-knowledge proof that the sum of the committed inputs is greater than the sum of the committed outputs, and that all the outputs are positive, namely they lie in the interval $[0, 2^n]$, where 2^n is much smaller than the group size. All current implementations of confidential transactions [Max16, MP15, PBF+, NM+16] use range proofs over committed values, where the proof size is linear in n. These range proofs are the main contributor to the size of a confidential transaction. In current implementations, a confidential transaction with only two outputs and 32 bits of precision is 5.5kB bytes, of which 5.3kB are allocated to the range proof.

We show in Section 6 that Bulletproofs greatly improve on this, even for a single range proof. The logarithmic proof size additionally enables the prover to aggregate multiple range proofs, e.g. for multiple outputs, into a single short proof. With Bulletproofs, m range proofs are merely $O(\log(m))$ additional group elements over a single range proof. This is already useful for confidential transactions in their current form as most Bitcoin transactions have two or more outputs. It also presents an intriguing opportunity to aggregate multiple range proofs from different parties into one proof, as would be needed, for example, in a CoinJoin transaction [Max13]. To do so, we present an MPC protocol that lets parties efficiently combine proofs without compromising confidentiality.

Confidential transaction implementations are available in side-chains [PBF+], private blockchains [And17], and in the popular privacy-focused cryptocurrency Monero [NM+16]. All these implementations would benefit from Bulletproofs.

At the time of writing, Bitcoin has roughly 50 million UTXOs from 22 million transactions (see `statoshi.info`). Using a 52-bit representation of bitcoin that can cover all values from 1 satoshi up to 21 million bitcoins, this results in roughly 160GB of range proof data using the current systems. Using aggregated Bulletproofs, the range proofs for all UTXOs would take less than 17GB, about a factor 10 reduction in size.

Mimblewimble. Recently an improvement to confidential transactions, called Mimblewimble [Jed16, Poe], provides further savings.

Jedusor [Jed16] realized that a Pedersen commitment to 0 can be viewed as a public key and that for a valid confidential transaction the difference between outputs, inputs and transaction fees must be 0. A prover constructing a confidential transaction can therefore sign the transaction with the difference of the outputs and inputs as the public key. This small change removes the need for a scriptSig which greatly simplifies the structure of confidential transactions. Poelstra [Poe] further refined and improved Mimblewimble and showed that these improvements enable a greatly simplified blockchain in which all spent transactions can be pruned and new nodes can efficiently validate the entire blockchain without downloading any old and spent transactions. Using aggregatable signatures, the blockchain can be compressed to a small subset of the block-headers as well as the remaining unspent transaction outputs and

the accompanying range proofs. Mimblewimble also allows transactions to be aggregated before sending them to the blockchain.

A Mimblewimble blockchain only grows with the size of the UTXO set. Using Bulletproofs, it would only grow with the number of transactions that have unspent outputs. Overall, Bulletproofs can not only act as a drop-in replacement for the range proofs in confidential transactions, but it can also help make Mimblewimble a practical scheme with a blockchain that is significantly smaller than the current Bitcoin blockchain.

1.2.2 Provisions

Dagher et al. [DBB+15] introduced the Provisions protocol which allows Bitcoin exchanges to prove that they are solvent without revealing any additional information. The protocol crucially relies on range proofs to prevent an exchange from inserting fake accounts with negative balances. These range proofs, which take up over 13GB, are the main contributors to the proof sizes of almost 18GB for a large exchange with 2 million customers. The proof size is in fact linear in the number of customers. Since in this protocol, one party (the exchange) has to construct many range proofs at once, the general Bulletproofs protocol from Section 4.3 is a natural replacement for the NIZK proof used in Provisions. With the proof size listed in Section 6, we obtain that the range proofs would take up less than 2kB with our protocol. Additionally, the other parts of the proof could be similarly compressed using the protocol from Section 5. The proof would then be dominated by one commitment per customer, with size 62 MB. This is roughly 300 times smaller then the current implementation.

1.2.3 Verifiable shuffles

Consider two lists of committed values $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$. The goal is to prove that the second list is a permutation of the first. This problem is called a *verifiable shuffle*. It has many applications in voting [FS01, Nef01], mix-nets [Cha82], and solvency proofs [DBB+15]. Neff [Nef01] gave a practical implementation of a verifiable shuffle and later work improved on it [Gro03, GI08a]. Currently the most efficient shuffle [BG12] has size $\sqrt{n}$.

Bulletproofs can be used to create a verifiable shuffle of size $O(\log n)$. The two lists of commitments are given as inputs to the circuit protocol from Section 5. The circuit can implement a shuffle by sorting the two lists and then checking that they are equal. A sorting circuit can be implemented using $O(n \cdot \log(n))$ multiplications which means that the proof size will be only $O(\log(n))$. This is much smaller than previously proposed protocols. Given the concrete efficiency of Bulletproofs, a verifiable shuffle using Bulletproofs would be very efficient in practice.

1.2.4 NIZK Proofs for Smart Contracts

The Ethereum [Woo14] system uses highly expressive smart contracts to enable complex transactions. Smart contracts, like any other blockchain trans-

action, are public and provide no inherent privacy. To bring privacy to smart contracts, non-interactive zero-knowledge (NIZK) proofs have been proposed as a tool to enable complex smart contracts that do not leak the user inputs [KMS+16, MSH17]. However, these protocols are limited as the NIZK proof itself is not suitable for verification by a smart contract. The reason is that communication over the blockchain with a smart contract is expensive, and the smart contract's own computational power is highly limited. SNARKs, which have succinct proofs and efficient verifiers, seem like a natural choice, but current practical SNARKs [BSCG+13] require a complex trusted setup. The resulting common reference strings (CRS) are long, specific to each application, and possess trapdoors. In Hawk [KMS+16], for instance, a different CRS is needed for each smart contract, and either a trusted party is needed to generate it, or an expensive multi-party computation is needed to distribute the trust among a few parties. On the other hand, for small applications like boardroom voting, one can use classical sigma protocols [MSH17], but the proof-sizes and expensive verification costs are prohibitive for more complicated applications.

Bulletproofs improves on this by enabling small proofs which do not need a trusted setup. The Bulletproofs verifier is still expensive, but there are multiple ways to work around this. First, a smart contract may act optimistically and only verify a proof if some party challenges its validity. Incentives can be used to ensure that rational parties never create an incorrect proof nor challenge a correct proof. This can be further improved by using an interactive referee delegation model [CRR11], previously proposed for other blockchain applications [BGB17, TR]. In this model, the prover provides a proof along with a succinct commitment to the verifier's execution trace. A challenger that disagrees with the computation also commits to his computation trace and the two parties engage in an interactive binary search to find the first point of divergence in the computation. The smart contract can then execute this single computation step and punish the party which provided a faulty execution trace. The intriguing property of this protocol is that even when a proof is challenged, the smart contract only needs to execute a single computation step, i.e. a single gate of the verification circuit. In combination with small Bulletproofs, this can enable more complex but privacy preserving smart contracts. Like other applications these NIZK proofs would benefit from the MPC protocol that we present in Section 4.5. For example, in an auction contract, where each bidder needs to prove some properties of their bids, the protocol can be used to combine the Bulletproofs into a single proof. Furthermore, the proof will hide which bidder submitted which bid.

1.2.5 Short Non-Interactive Protocols for Arithmetic Circuits without a Trusted Setup

Non-interactive zero-knowledge protocols for general statements are not possible without using a common reference string, which should be known by both the prover and the verifier. Many efficient non-interactive zero-knowledge proofs and arguments for arithmetic circuit satisfiability have been developed

[Mic94, KP95, GS08, BSCG+13], and highly efficient protocols are known. However, aside from their performance, these protocols differ in the complexity of their common reference strings. Some, such as those in [BSCG+13], are highly structured, and sometimes feature a trapdoor, while some are simply chosen uniformly at random. Security proofs assume that the common reference string was honestly generated. In practice, the common reference string can be generated by a trusted third party, or using a secure multi-party computation protocol, in which case, uniformly-random common reference strings are significantly easier to generate. This helps to alleviate concerns about embedded trapdoors, as with the trusted setup ceremony used to generate the public parameters for [BSCG+14].

Zero-knowledge SNARKs have been the subject of extensive research [Gro10, BCCT12,GGPR13,BCCT13,PHGR16,BSCG+13,Gro16]. They generate constant-sized proofs for any statement, and have extremely fast verification time. However, they have highly complex common reference strings which require lengthy and computationally intensive protocols [BGG17] to generate distributively. They also rely on strong unfalsifiable assumptions such as the knowledge-of-exponent assumption.

Examples of non-interactive protocols that do not require a trusted setup include [Mic94, BCC+16, BCG+17b, BSBC+17].

Ben-Sasson et al. present a proof system [BCG+17a] and implementation [BSBC+17] called Scalable Computational Integrity (SCI). While SCI has a simple setup, and relies only on collision-resistant hash functions, the system is not zero-knowledge and still experiences worse performance than [BSCG+13, BCC+16]. The proof sizes are roughly 42 MB large in practice. In subsequent work Ben-Sasson presented STARKs [BSBTHR17], which are zero-knowledge and more efficient than SCI. However even with these improvements the proof size is still 1.8MB and constructing a proof is very costly (about 7 minutes and 130 GB of RAM for a circuit of size 2^{16}).

In later work, Bootle et al. [BCG+17b] present an argument for arithmetic circuit satisfiability based on collision-resistant hash functions. The cost for the prover scales linearly in the circuit size, and the verifier is slightly sub-linear, meaning that the protocol is asymptotically more efficient than [BCC+16, BCG+17a]. However, while sub-linear, the proof size is much larger, scaling with the square root of the circuit size, and furthermore, the computational costs hide large constants which prevent the argument from being practical.

1.3 Additional Related Work

Much of the research related to electronic payments that predates Bitcoin [Nak08] focused on efficient anonymous and confidential payments [CHL05, Cha82] . With the advent of blockchain-based cryptocurrencies, the question of privacy and confidentiality in transactions has gained a new relevance. While the original Bitcoin paper [Nak08] claimed that Bitcoin would provide anonymity through pseudonymous addresses early work on Bitcoin showed that the anonymity is limited [MPJ+13, AKR+13]. Given these limitations various methods have

been proposed to help improve the privacy of Bitcoin transactions. CoinJoin [Max13], proposed by Maxwell, allows users to hide information about the amounts of transactions by merging two or more transactions. This ensures that among the participants who join their transactions, it is impossible to tell which transaction inputs correspond to which transaction outputs. However, users do require some way of searching for other users, and furthermore, should be able to do so without relying on a trusted third party. CoinShuffle [RMSK14] tried to fulfill this requirement by taking developing the ideas of CoinJoin and proposing a new Bitcoin mixing protocol which is completely decentralized.

Monero [Mon] is a cryptocurrency which employs cryptographic techniques to achieve strong privacy guarantees. These include stealth addresses, ring-signatures [vS13], and ring confidential transactions [NM$^+$16]. Range proofs are proofs that a secret value, which has been encrypted or committed to, lies in a certain interval. Range proofs do not leak any information about the secret value, other than the fact that they lie in the interval. Lipmaa [Lip03] presents a range proof which uses integer commitments, and Lagrange's four-square theorem which states that every positive integer y can be expressed as a sum of four squares. Groth [Gro05] notes that the argument can be optimized by considering $4y + 1$, since integers of this form only require three squares. The 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. Additionally, a trusted setup is required to generate the RSA modulus or a prohibitively large modulus needs to be used [San99]. Camenisch et al. [CCS08] use a different approach. The verifier provides signatures on a small set of digits. The prover commits to the digits of the secret value, and then proves in zero-knowledge that the value matches the digits, and that each commitment corresponds to one of the signatures. They show that their scheme can be instantiated securely using both RSA accumulators [BdM93] and the weak Boneh-Boyen signature scheme [BB04]. However, these range proofs require a trusted setup. Approaches based on the n-ary digits of the secret value are limited to proving that the secret value is in an interval of the form $\left[0, n^k - 1\right]$. One can produce range proofs for more general intervals by using homomorphic commitments to translate intervals, and by using a combination of two different range proofs to conduct range proofs for intervals of different widths. However, [CLS10] presented an alternative digital decomposition which enables an interval of general width to be handled using a single range proof.

2 Preliminaries

Before we present Bulletproofs, we first review some of the underlying tools.

2.1 Commitments

Definition 1 (Commitment)**.** *A non-interactive commitment scheme consists of a pair of probabilistic polynomial time algorithms* $(\mathrm{Setup}, \mathrm{Com})$*. The setup*

algorithm $\text{pp} \leftarrow \text{Setup}(1^\lambda)$ *generates public parameters* pp *for the scheme, for security parameter* λ*. The commitment algorithm* Com_{pp} *defines a function* $\mathsf{M}_{\text{pp}} \times \mathsf{R}_{\text{pp}} \to \mathsf{C}_{\text{pp}}$ *for message space* M_{pp}*, randomness space* R_{pp} *and commitment space* C_{pp} *determined by* pp*. For a message* $x \in \mathsf{M}_{\text{pp}}$*, the algorithm draws* $r \xleftarrow{\$} \mathsf{R}_{\text{pp}}$ *uniformly at random, and computes commitment* $\mathbf{com} = \text{Com}_{\text{pp}}(x; r)$*.*

Definition 2 (Homomorphic Commitments)**.** *A homomorphic commitment scheme is a non-interactive commitment scheme such that* $\mathsf{M}_{\text{pp}}, \mathsf{R}_{\text{pp}}$ *and* C_{pp} *are all abelian groups, and for all* $x_1, x_2 \in \mathsf{M}_{\text{pp}}$*,* $r_1, r_2 \in \mathsf{R}_{\text{pp}}$*, 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 3 (Hiding Commitment)**.** *A commitment scheme is said to be hiding if for all PPT adversaries* $\mathsf{A}_{Hiding,1^\lambda}$ *there exists a negligible function* $\mu(\lambda)$ *such that.*

$$\left| \text{P}\left[b = b' \;\middle|\; \begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda); \\ (x_0, x_1) \in \mathsf{M}_{\text{pp}}^2 \leftarrow \mathsf{A}_{Hiding,1^\lambda}(\text{pp}), \\ b \xleftarrow{\$} \{0,1\}, r \xleftarrow{\$} \mathsf{R}_{\text{pp}}, \\ \mathbf{com} = \text{Com}(x_b; r), \\ b' \leftarrow \mathsf{A}_{Hiding,1^\lambda}(\text{pp}, \mathbf{com}) \end{array} \right] - \frac{1}{2} \right| \leqslant \mu(\lambda)$$

where the probability is over b, r, Setup *and* $\mathsf{A}_{Hiding,1^\lambda}$*. If* $\mu(\lambda) = 0$ *then we say the scheme is* perfectly hiding*.*

Definition 4 (Binding Commitment)**.** *A commitment scheme is said to be binding if for all PPT adversaries* $\mathsf{A}_{Hiding,1^\lambda}$ *there exists a negligible function* μ *such that.*

$$\text{P}\left[\begin{array}{l} \text{Com}(x_0; r_0) \\ = \text{Com}(x_1; r_1) \end{array} \;\middle|\; \begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda), \\ x, x', r, r' \leftarrow \mathsf{A}_{Binding,1^\lambda}(\text{pp}) \end{array} \right] \leqslant \mu(\lambda)$$

where the probability is over Setup *and* $\mathsf{A}_{Hiding,1^\lambda}$*. If* $\mu(\lambda) = 0$ *then we say the scheme is* perfectly binding*.*

We will drop the security parameter λ from the notation when it is implicit. In particular, note that the order p of the groups used is implicitly dependent on 1^λ.

Definition 5 (Pedersen Commitment)**.** $\mathsf{M}_{\text{pp}}, \mathsf{R}_{\text{pp}} = \mathbb{Z}_p$*,* $\mathsf{C}_{\text{pp}} = \mathbb{G}$ *of order* p*.*
$\text{Setup} : g, h \xleftarrow{\$} \mathbb{G}$
$\text{Com}(x; r) = (g^x h^r)$

Definition 6 (Pedersen Vector Commitment)**.** $\mathsf{M}_{\text{pp}} = \mathbb{Z}_p^n$*,* $\mathsf{R}_{\text{pp}} = \mathbb{Z}_p$*,* $\mathsf{C}_{\text{pp}} = \mathbb{G}^n$ *with* $\mathbb{G}$ *of order* p
$\text{Setup} : \mathbf{g} = (g_1, \ldots, g_n), h \xleftarrow{\$} \mathbb{G}$
$\text{Com}(\mathbf{x}; r) = h^r \mathbf{g}^{\mathbf{x}} = h^r \prod_i g_i^{x_i}$

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

2.2 Zero-Knowledge Arguments of Knowledge

In this paper the common reference string will always be a public key ck for the Pedersen commitment scheme.

We will consider arguments consisting of three interactive algorithms $(\mathcal{K}, \mathcal{P}, \mathcal{V})$, all running in probabilistic polynomial time. These are the common reference string generator $\mathcal{K}$, the prover $\mathcal{P}$, and the verifier $\mathcal{V}$. On input 1^λ, algorithm $\mathcal{K}$ produces a common reference string σ. The transcript produced by $\mathcal{P}$ and $\mathcal{V}$ when interacting on inputs s and t is denoted by $tr \leftarrow \langle \mathcal{P}(s), \mathcal{V}(t) \rangle$. We write $\langle \mathcal{P}(s), \mathcal{V}(t) \rangle = b$ depending on whether the verifier rejects, $b = 0$, or accepts, $b = 1$.

Let $\mathcal{R} \subset \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ be a polynomial-time-decidable ternary relation. Given σ, we call w a witness for a statement u if $(\sigma, u, w) \in \mathcal{R}$, and define the CRS-dependent language

$$\mathcal{L}_\sigma = \{x \mid \exists w : (\sigma, x, w) \in \mathcal{R}\}$$

as the set of statements x that have a witness w in the relation $\mathcal{R}$.

Definition 7 (Argument of Knowledge)**.** *The triple $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ is called an* argument of knowledge *for relation $\mathcal{R}$ if it satisfies the following definitions.*

Definition 8 (Perfect completeness)**.** *$(\mathcal{P}, \mathcal{V})$ has perfect completeness if for all non-uniform polynomial time adversaries $\mathcal{A}$*

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

Definition 9 (Computational Witness-Extended Emulation)**.** *$(\mathcal{P}, \mathcal{V})$ has* computational witness-extended emulation *if for all deterministic polynomial time $\mathcal{P}^*$ there exists an expected polynomial time emulator $\mathcal{E}$ such that for all interactive adversaries $\mathcal{A}$*

$$\begin{aligned} &\mathrm{P}\left[\mathcal{A}(tr) = 1 \,\middle|\, \begin{array}{l} \sigma \leftarrow \mathcal{K}, (u, s) \leftarrow \mathcal{A}(\sigma), \\ tr \leftarrow \langle \mathcal{P}^*(\sigma, u, s), \mathcal{V}(\sigma, u) \rangle \end{array}\right] \\ \approx\ &\mathrm{P}\left[\begin{array}{l|l} \mathcal{A}(tr) = 1 & \sigma \leftarrow \mathcal{K}, (u, s) \leftarrow \mathcal{A}(\sigma), \\ \wedge (tr \text{ is accepting} & (tr, w) \leftarrow \mathcal{E}^{\mathcal{O}}(\sigma, u) \\ \implies (\sigma, u, w) \in \mathcal{R}) & \end{array}\right] \end{aligned}$$

where the oracle is given by $\mathcal{O} = \langle \mathcal{P}^*(\sigma, u, s), \mathcal{V}(\sigma, u) \rangle$, 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 $\mathcal{A}$.*

We use witness-extended emulation to define knowledge-soundness as used for example in [BCC+16] and defined in [GI08b,Lin03]. Informally, whenever an adversary produces an argument which satisfies the verifier with some probability, then there exists an emulator producing an identically distributed argument

with the same probability, but also a witness. The value s can be considered to be the internal state of $\mathcal{P}^*$, including randomness. The emulator is permitted to rewind the interaction between the prover and verifier to any move, and resume with the same internal state for the prover, but with fresh randomness for the verifier. Whenever $\mathcal{P}^*$ makes a convincing argument when in state s, $\mathcal{E}$ can extract a witness, and therefore, we have an argument of knowledge of w such that $(\sigma, u, w) \in \mathcal{R}$.

Definition 10 (Public Coin)**.** *An argument $(\mathcal{P}, \mathcal{V})$ is called* public coin *if all messages sent from the verifier to the prover are chosen uniformly at random and independently of the prover's messages, i.e., the challenges correspond to the verifier's randomness ρ.*

An argument is zero knowledge if it does not leak information about w apart from what can be deduced from the fact that $(\sigma, x, w) \in \mathcal{R}$. We will present arguments that have special honest-verifier zero-knowledge. This means that given the verifier's challenge values, it is possible to efficiently simulate the entire argument without knowing the witness.

Definition 11 (Perfect Special Honest-Verifier Zero-Knowledge)**.** *A public coin argument $(\mathcal{P}, \mathcal{V})$ is a* perfect special honest verifier zero knowledge (SHVZK) *argument for R if there exists a probabilistic polynomial time simulator $\mathcal{S}$ such that for all interactive non-uniform polynomial time adversaries $\mathcal{A}$*

$$\Pr\left[\begin{array}{l|l} (\sigma,u,w)\in\mathcal{R} & \sigma\leftarrow\mathcal{K}(1^\lambda),(u,w,\rho)\leftarrow\mathcal{A}(\sigma),\\ \text{and } \mathcal{A}(tr)=1 & tr\leftarrow\langle\mathcal{P}(\sigma,u,w),\mathcal{V}(\sigma,u;\rho)\rangle \end{array}\right]$$

$$=\Pr\left[\begin{array}{l|l} (\sigma,u,w)\in\mathcal{R} & \sigma\leftarrow\mathcal{K}(1^\lambda),(u,w,\rho)\leftarrow\mathcal{A}(\sigma),\\ \text{and } \mathcal{A}(tr)=1 & tr\leftarrow\mathcal{S}(u,\rho) \end{array}\right]$$

where ρ is the public coin randomness used by the verifier.

2.2.1 A General Forking Lemma.

We briefly describe the forking lemma of [BCC+16].

Suppose that we have a $(2\mu+1)$-move public-coin argument with μ challenges, $x_1, \ldots, x_\mu$ in sequence. Let $n_i \geqslant 1$ for $1 \leqslant i \leqslant \mu$. Consider $\prod_{i=1}^{\mu} n_i$ accepting transcripts with challenges in the following tree format. The tree has depth μ and $\prod_{i=1}^{\mu} n_i$ leaves. The root of the tree is labeled with the statement. Each node of depth $i < \mu$ has exactly n_i children, each labeled with a distinct value of the ith challenge x_i.

This can be referred to as an $(n_1, \ldots, n_\mu)$-tree of accepting transcripts. Given a suitable tree of accepting transcripts, one can compute a valid witness for our inner-product argument, range proof, and argument for arithmetic circuit satisfiability. This is a natural generalization of special-soundness for Sigma-protocols, where $\mu = 1$ and $n = 2$. Combined with Theorem 1, this shows that the protocols have witness-extended emulation, and hence, the prover cannot produce an accepting transcript unless they know a witness. For simplicity in

the following lemma, we assume that the challenges are chosen uniformly from $\mathbb{Z}_p$ where $|p| = \lambda$, but any sufficiently large challenge space would suffice.

Theorem 1 (Forking Lemma, [BCC+16])**.** *Let $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ be a $(2\mu + 1)$-move, public coin interactive protocol. Let χ be a witness extraction algorithm that always succeeds in extracting a witness from an $(n_1, \ldots, n_\mu)$-tree of accepting transcripts in probabilistic polynomial time. Assume that $\prod_{i=1}^{\mu} n_i$ is bounded above by a polynomial in the security parameter λ. Then $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ has witness-extended emulation.*

We now define range proofs, which are proofs that the prover knows an opening to a commitment, such that the committed value is in a certain range. Range proofs can, for instance, be used to show that an integer commitment is to a positive number or that two homomorphic commitments to elements in a field of prime order will not overflow modulo the prime when they are added together.

Definition 12 (Zero-Knowledge Range Proof)**.** *Given a commitment scheme* $(\mathrm{Setup}, \mathrm{Com})$ *over a message space* M_{pp} *which is a set with a total ordering, a Zero-Knowledge Range Proof is a protocol for the following relation:* $\{(1^\lambda, \mathrm{pp}, \mathbf{com} \in \mathsf{C}_{\mathrm{pp}}, l, r \in \mathsf{M}_{\mathrm{pp}}; \mathrm{pp} = \mathrm{Setup}(1^\lambda) \wedge \mathbf{com} = \mathrm{Com}(x; r) \wedge x \geqslant l \wedge x \leqslant r\}$

2.3 Notation

Let $\mathbb{G}$ denote a cyclic group of prime order p, and let $\mathbb{Z}_p$ denote the ring of integers modulo p. Let $\mathbb{G}^n$ and $\mathbb{Z}_p^n$ be vector spaces of dimension n over $\mathbb{G}$ and $\mathbb{Z}_p$ respectively. Let $\mathbb{Z}_p^\star$ denote $\mathbb{Z}_p \backslash \{0\}$. Generators of $\mathbb{G}$ are denoted by $g, h, v, u \in \mathbb{G}$. Group elements which represent commitments are capitalized and blinding factors are denoted by Greek letters, i.e. $C = g^a h^\alpha \in \mathbb{G}$ is a Pedersen commitment to a. If not otherwise clear from context $x, y, z \in \mathbb{Z}_p^\star$ are uniformly distributed challenges. Throughout the paper, we will also be using vector notations defined as follows. Bold font denotes vectors, i.e. $\mathbf{a} \in \mathbb{F}^n$ is a vector with elements $a_1, \ldots, a_n \in \mathbb{F}$. Capitalized bold font denotes matrices, i.e. $\mathbf{A} \in \mathbb{F}^{n \times m}$ is a matrix with n rows and m columns. Such that $a_{i,j}$ is the element of $\mathbf{A}$ in the ith row and jth column. For a scalar $c \in \mathbb{Z}_p$ and a vector $\mathbf{a} \in \mathbb{Z}_p^n$, we denote by $\mathbf{b} = c \cdot \mathbf{a} \in \mathbb{Z}_p^n$ the vector where $b_i = c \cdot a_i$. Furthermore, let $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^n a_i \cdot b_i$ denote the inner product between two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}^n$, and $\mathbf{a} \circ \mathbf{b} = (a_1 \cdot b_1, \ldots, a_n \cdot b_n) \in \mathbb{F}^n$ the Hadamard product or entry wise multiplication of two vectors. The matrix product between a matrix $\mathbf{A} \in \mathbb{Z}_p^{n \times m}$ and a vector $\mathbf{b} \in \mathbb{Z}_p^m$ is denoted as $\mathbf{A} \cdot \mathbf{b} = \mathbf{c} \in \mathbb{Z}_p^n$ with $c_i = \sum_{j=1}^m a_{i,j} \cdot b_j \in \mathbb{Z}_p$. Note that the product is not commutative, i.e. for $\mathbf{c} \in \mathbb{Z}_p^n$ and $\mathbf{d} \in \mathbb{Z}_p^m$: $\mathbf{c} \cdot \mathbf{A} = \mathbf{d} \in \mathbb{Z}_p^m$ such that $d_j = \sum_{i=1}^n c_i \cdot a_{i,j} \in \mathbb{Z}_p$.
Further we extend the vector notation to Pedersen vector commitments. Specifically let $\mathbf{g} = (g_1, \ldots, g_n) \in \mathbb{G}^n$ be a vector of generators then $C = \mathbf{g}^{\mathbf{a}} = \prod_{i=1}^n g_i^{a_i}$ is a commitment to vector $\mathbf{a} \in \mathbb{Z}_p^n$. Given such a commitment C and a vector $\mathbf{b} \in \mathbb{Z}_p^n$ with non-zero entries, it is possible to view C as a new commitment to

$\mathbf{a} \circ \mathbf{b}$. This is done by defining $g'_i = g_i^{b_i^{-1}}$ such that $C = \prod_{i=1}^{n} g_i'^{a_i \cdot b_i}$. The binding property of this new commitment holds if the old commitment was binding.
Let $\mathbf{a}||\mathbf{b}$ denote the concatenation of two vectors, i.e. for $a \in \mathbb{Z}_p^n$ and $b \in \mathbb{Z}_p^m$ $\mathbf{c} = \mathbf{a}||\mathbf{b} \in \mathbb{Z}_p^{n+m}$. We also define subsets of vectors using a pythonic notation, i.e. $\mathbf{a}_{[:k]} = (a_1, \ldots, a_k) \in \mathbb{F}^k$ or $\mathbf{a}_{[k:]} = (a_{k+1}, \ldots, a_n) \in \mathbb{F}^{n-k}$. We use $\mathbf{k}^n$ to denote the vector containing the first n powers of $k \in \mathbb{Z}_p^\star$, i.e. $\mathbf{k}^n = (1, k, k^2, \ldots, k^{n-1}) \in \mathbb{Z}_p^{\star n}$ or $\mathbf{2}^n = (1, 2, 4, \ldots, 2^{n-1})$. Equivalently $\mathbf{k}^{-n} = (\mathbf{k^{-1}})^n = (1, k^{-1}, \ldots, k^{-n+1})$.
We define zero knowledge proofs for relations using the following notation
$\{(\text{Public Input}; \text{Witness}) : \text{Relation}\}$

3 Improved Inner-Product Argument

Bootle et al. [BCC+16] introduced an efficient inner-product argument and show how it can be leveraged to construct zero-knowledge proofs for arithmetic circuit satisfiability with low communication complexity. The argument is an argument of knowledge of openings to two Pedersen vector commitments satisfying an inner product relation. We demonstrate how the communication complexity of the argument can be further reduced from $6\log_2(n)$ in [BCC+16] to only $2\log_2(n)$ where n is the size of the two vectors. To achieve this improvement, we modify the relation that is proved. We then explain how this protocol can be used to construct an efficient range proof and a proof system for arbitrary arithmetic circuits in Sections 4 and 5.

The protocol takes as input a Pedersen vector commitment (see Definition 6) to the two vectors $\mathbf{a}$ and $\mathbf{b}$ as well as c and proves that $c = \langle \mathbf{a}, \mathbf{b} \rangle$. The logarithmic complexity is achieved by halving the size of the vector base for the commitment in every iteration. Consider the following observation: For independent generators g_1, g_2, $P = g_1^a g_2^b$ is a binding commitment to a, b. If a verifier, using a random challenge x, could construct $P' = g'^{a \cdot x + b \cdot x^{-1}}$ from P then P' would still be a commitment to a, b but now the prover can demonstrate knowledge of a, b by producing $a \cdot x + b \cdot x^{-1}$ which is half the size in $\mathbb{Z}_p$. So how can a verifier produce P' from P? Let us first set $g' = g_1^{x^{-1}} g_2^x$ then we get that $P' = g'^{a \cdot x + b \cdot x^{-1}} = g_1^{a + b \cdot x^{-2}} g_2^{b + a \cdot x^2} = P \cdot (g_1^b)^{x^{-2}} (g_2^a)^{x^2}$. So, if the prover along with P also sends $R = g_1^b$ and $L = g_2^a$ then we get $P' = L^{x^2} \cdot P \cdot R^{x^{-2}}$. Interestingly the technique works if P is a commitment to a much larger vector but the commitments L, R and P are always just a single group element. Additionally, we show in Protocol 1 that this can also be done for two vectors in parallel such that the inner product of the two vectors does only changes by a correction factor which the verifier again can compute himself from the challenge.
More formally, the input to the inner product argument are independent generators $\mathbf{g}, \mathbf{h} \in \mathbb{G}^n$ a scalar $c \in \mathbb{Z}_p$ and the commitment P such that $P = \mathbf{g^a h^b}$. The argument demonstrates that $\langle \mathbf{a}, \mathbf{b} \rangle = c$ given pairwise hardness of computing discrete logarithm relations between each pair of two distinct group elements from $\mathbf{g}, \mathbf{h}$. We assume w.l.o.g. that n is a power of 2, since this can always be

achieved by padding $\mathbf{g}, \mathbf{h}, \mathbf{a}$, and $\mathbf{b}$. Protocol 2 is, therefore, an efficient proof system for the following relation.

$$\begin{aligned} &\{(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, u, P \in \mathbb{G}, c \in \mathbb{Z}_p; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n) : \\ &P = \mathbf{g}^{\mathbf{a}}\mathbf{h}^{\mathbf{b}} \wedge c = \langle \mathbf{a}, \mathbf{b} \rangle\} \end{aligned} \tag{1}$$

The protocol requires the communication of $2 \cdot \lceil \log_2(n) \rceil$ elements in $\mathbb{G}$ as well as 2 elements in $\mathbb{Z}_p$. The prover's work is dominated by $4 \cdot n$ group exponentiations. The verifier's work by $2 \cdot n$ exponentiations. For more details on our implementation see Section 6. We split the protocol into two parts. First we show a proof system for the following relation in Protocol 1:

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

Then we show that we can use our proof system for Relation (2) to build a new proof system for Relation (1) using Protocol 2.

Theorem 2 (Inner-Product Argument)**.** *The argument presented in Protocol 1 and 2 has perfect completeness and statistical witness-extended-emulation for either extracting a non-trivial discrete logarithm relation or a valid witness.*

The proof for Theorem 2 is in Appendix A.

4 Range Proof Protocol with Logarithmic Size

We now present a novel protocol for conducting short and aggregatable range proofs. The protocol makes uses of an improved version of the inner product argument from Protocol 2. First, in Section 4.1, we describe how to construct a range proof that requires the verifier to check an inner product between two vectors. Then, in Section 4.2, we show that this check can be replaced with an efficient inner-product argument. In Section 4.3, we show how to efficiently prove knowledge of m numbers that are all within a given range.
In Section 4.4, we discuss how interactive public coin protocols can be made non-interactive by using the Fiat-Shamir heuristic, in the random oracle model. In Section 4.5 we present an efficient MPC protocol that allows multiple parties to construct a single aggregate range proof. Finally, in Section 4.6, we discuss an extension that enables a switch to quantum-secure range proofs in the future.

4.1 Inner-Product Range Proof

We present a protocol which uses the improved inner-product argument to construct a range proof. The proof convinces the verifier that a commitment V contains a number v that is in a certain range, without revealing v. Bootle et al. [BCC+16] presents a proof system for arbitrary arithmetic circuits, and in Section 5, we demonstrate that our improvements to the inner product argument also transfer to this proof system. It is of course possible to prove that a commitment is in a given range using an arithmetic circuit and asymptotically [BCC+16] could be used to construct logarithmically (in the length of v)

Input: $(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n, u, P \in \mathbb{G}; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n)$ (3)

$\mathcal{P}_{\mathsf{IP}}$'s input: $(\mathbf{g}, \mathbf{h}, u, P, \mathbf{a}, \mathbf{b})$ (4)

$\mathcal{V}_{\mathsf{IP}}$'s input: $(\mathbf{g}, \mathbf{h}, u, P)$ (5)

Output: $\{\mathcal{V}_{\mathsf{IP}}$ accepts or $\mathcal{V}_{\mathsf{IP}}$ rejects$\}$ (6)

if $n = 1$: (7)

$\mathcal{P}_{\mathsf{IP}} \rightarrow \mathcal{V}_{\mathsf{IP}} : a, b$ (8)

$c = a \cdot b$ (9)

$\mathcal{V}_{\mathsf{IP}}$ checks if $P = g^a h^b u^c$: (10)

if yes, $\mathcal{V}_{\mathsf{IP}}$ accepts (11)

otherwise, $\mathcal{V}_{\mathsf{IP}}$ rejects (12)

else: $(n > 1)$ (13)

$\mathcal{P}_{\mathsf{IP}}$ computes: (14)

$$n' = \frac{n}{2} \tag{15}$$

$$c_L = \langle \mathbf{a}_{[:n']}, \mathbf{b}_{[n':]} \rangle \in \mathbb{Z}_p \tag{16}$$

$$c_R = \langle \mathbf{a}_{[n':]}, \mathbf{b}_{[:n']} \rangle \in \mathbb{Z}_p \tag{17}$$

$$L = \mathbf{g}_{[n':]}^{\mathbf{a}_{[:n']}} \mathbf{h}_{[:n']}^{\mathbf{b}_{[n':]}} u^{c_L} \tag{18}$$

$$R = \mathbf{g}_{[:n']}^{\mathbf{a}_{[n':]}} \mathbf{h}_{[n':]}^{\mathbf{b}_{[:n']}} u^{c_R} \tag{19}$$

$\mathcal{P}_{\mathsf{IP}} \rightarrow \mathcal{V}_{\mathsf{IP}} : L, R$ (20)

$\mathcal{V}_{\mathsf{IP}} : x \xleftarrow{\$} \mathbb{Z}_p^\star$ (21)

$\mathcal{V}_{\mathsf{IP}} \rightarrow \mathcal{P}_{\mathsf{IP}} : x$ (22)

$\mathcal{P}_{\mathsf{IP}}$ and $\mathcal{V}_{\mathsf{IP}}$ compute: (23)

$$\mathbf{g}' = \mathbf{g}_{[:n']}^{x^{-1}} \circ \mathbf{g}_{[n':]}^{x} \tag{24}$$

$$\mathbf{h}' = \mathbf{h}_{[:n']}^{x} \circ \mathbf{h}_{[n':]}^{x^{-1}} \tag{25}$$

$$P' = L^{x^2} P R^{x^{-2}} \tag{26}$$

$\mathcal{P}_{\mathsf{IP}}$ computes: (27)

$$\mathbf{a}' = \mathbf{a}_{[:n']} \cdot x + \mathbf{a}_{[n':]} \cdot x^{-1} \in \mathbb{Z}_p^{n'} \tag{28}$$

$$\mathbf{b}' = \mathbf{b}_{[:n']} \cdot x^{-1} + \mathbf{b}_{[n':]} \cdot x \in \mathbb{Z}_p^{n'} \tag{29}$$

recursively run Protocol 1 on input (30)

$(\mathbf{g}', \mathbf{h}', u, P'; \mathbf{a}', \mathbf{b}')$ (31)

Protocol 1: Improved Inner-Product Argument

$$\mathcal{V}_{\mathsf{IP}} : x \xleftarrow{\$} \mathbb{Z}_p^\star \tag{32}$$
$$\mathcal{V}_{\mathsf{IP}} \to \mathcal{P}_{\mathsf{IP}} : x \tag{33}$$
$$P' = P \cdot u^{x \cdot c} \tag{34}$$
$$\text{Run Protocol 1 on Input } (\mathbf{g}, \mathbf{h}, u^x, P'; \mathbf{a}, \mathbf{b}) \tag{35}$$

Protocol 2: Proof system for Relation (1) using Protocol 1.

sized range proofs.
However, the circuit would need to implement the commitment function, e.g. a multi-exponentiation for Pedersen commitments, leading to a large and complex circuit.
We, therefore, demonstrate that we can construct a range proof more directly. The range proof takes advantage of the fact that if V is a Pedersen commitment, then it is an element in the same group that is used to perform the inner product argument. We extend this idea in Section 5 to show that our circuit can take an arbitrary number of commitments as input.
The proof system uses the homomorphic property of Vector Pedersen Commitments to construct commitments to two polynomials $l(X)$ and $r(X)$ in $\mathbb{Z}_p^n[X]$, i.e. the coefficients of $l(X)$ and $r(X)$ are vectors in $\mathbb{Z}_p^n$. Using these vector-polynomial commitments, the prover and verifier engage in an inner product argument to verifiably compute the inner product of $l(X)$ and $r(X)$. These polynomials are carefully constructed such that the zero-coefficient of $\langle l(X), r(X)\rangle \in \mathbb{Z}_p[X]$ has a special form if and only if v is in the range. This can be viewed as encoding the range proof circuit in the zero-coefficient of $\langle l(x), r(x)\rangle$. For simplicity, we describe the product as an interactive protocol where all the verifiers messages are random elements in $\mathbb{Z}_p$. As discussed in Section 4.4, this protocol can be turned into a non-interactive range proof using the Fiat-Shamir heuristic. In Section 4.2 we show how to use the inner product argument to turn the range proof into a highly efficient proof whose size only grows logarithmically in the bits of the range proven.
Formally, let v be a number in $[0, 2^n - 1]$ with $n = O(\lambda)$, and V be a commitment to v using randomness γ. Let $\mathbf{a} = (a_1, \ldots, a_n)$ be the vector containing the bits of v, so that $\langle \mathbf{a}, \mathbf{2}^n \rangle = v$. The prover $\mathcal{P}$ commits to $\mathbf{a}$ as well as blinding vectors $\mathbf{s}_L, \mathbf{s}_R$ using constant sized vector commitments. $\mathcal{P}$ then constructs the polynomial $t(X) \in \mathbb{Z}_p[X]$ as a function of $\mathbf{a}, \mathbf{s}_L, \mathbf{s}_R$ whose zero coefficient is independent of $\mathbf{a}$ if and only if $\mathbf{a}$ indeed contains only bits. $t(X)$ is exactly the inner product of $l(X), r(X) \in \mathbb{Z}_p^n[X]$. $l(X)$ and $r(X)$ are such that the Verifier $\mathcal{V}$ can himself construct a commitment to them.
Concretely the proof system proves the following relation which is equivalent to a range proof relation by Definition 12 using a Pedersen commitment scheme

and range $[0, 2^n - 1]$:

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

To prove the statement, $\mathcal{P}$ and $\mathcal{V}$ engage in the following zero knowledge protocol.

$$\mathbf{a}_L \in \{0,1\}^n \text{ s.t. } \langle \mathbf{a}_L, \mathbf{2}^n \rangle = v \tag{36}$$
$$\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^n \qquad \in \mathbb{Z}_p^n \tag{37}$$
$$\alpha \xleftarrow{\$} \mathbb{Z}_p \tag{38}$$
$$A = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R} \qquad \in \mathbb{G} \tag{39}$$
$$\mathbf{s}_L, \mathbf{s}_R \xleftarrow{\$} \mathbb{Z}_p^n \tag{40}$$
$$\rho \xleftarrow{\$} \mathbb{Z}_p \tag{41}$$
$$S = h^\rho \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R} \qquad \in \mathbb{G} \tag{42}$$
$$\mathcal{P} \rightarrow \mathcal{V} : A, S \tag{43}$$
$$\mathcal{V} : y, z \xleftarrow{\$} \mathbb{Z}_p^\star \tag{44}$$
$$\mathcal{V} \rightarrow \mathcal{P} : y, z \tag{45}$$

The prover now constructs the two degree 1 polynomials $l(X)$ and $r(X)$ in $\mathbb{Z}_p^n[X]$ and computes

$$t(X) = \langle l(X), r(X) \rangle \in \mathbb{Z}_p[X]$$

$\mathcal{V}$ can construct commitments to $l(X), r(X)$ from V, A, and S, as well as y and z.

$$\begin{aligned}
l(X) &= \mathbf{a}_L - z \cdot \mathbf{1}^n + \mathbf{s}_L \cdot X && \in \mathbb{Z}_p[X] \\
r(X) &= \mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n + \mathbf{s}_R \cdot X) + z^2 \cdot \mathbf{2}^n && \in \mathbb{Z}_p[X] \\
t(X) &= \langle l(X), r(X) \rangle = \sum_{i=0}^{2} t_i \cdot X^i && \in \mathbb{Z}_p[X] \\
t_0 &= \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n \rangle + z \cdot \langle \mathbf{a}_L - \mathbf{a}_R, \mathbf{y}^n \rangle \\
&\quad + z^2 \cdot \langle \mathbf{2}^n, \mathbf{a}_L \rangle + k(y, z) && \in \mathbb{Z}_p \\
k(y, z) &= -z^2 \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle - z^3 \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle && \in \mathbb{Z}_p
\end{aligned}$$

Note that if

$$\mathbf{a}_L = \mathbf{a}_R - \mathbf{1}^n \wedge \mathbf{a}_L \circ \mathbf{a}_R = \mathbf{0}^n \wedge \langle \mathbf{a}_L, \mathbf{2}^n \rangle = v \tag{46}$$

then

$$\begin{aligned}
t_0 &= z \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle + z^2 \cdot \langle \mathbf{a}_L, \mathbf{2}^n \rangle + k(y, z) \\
&= z \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle + z^2 \cdot v + k(y, z)
\end{aligned}$$

i.e. t_0 is a function of y, z and v. Furthermore, a commitment to t_0 can be constructed from y, z, and a homomorphic commitment to v. The proof of Theorem 1 shows if t_0 has this form, then (46) must hold. The prover therefore commits to t_1, t_2 using Pedersen commitments, and $\mathcal{P}$ and $\mathcal{V}$ engage in a polynomial identity testing protocol to show that $t(X) = \langle l(X), r(X) \rangle$.

$$\tau_1, \tau_2 \xleftarrow{\$} \mathbb{Z}_p \tag{47}$$

$$T_i = g^{t_i} h^{\tau_i} \quad i = \{1, 2\} \qquad \in \mathbb{G} \tag{48}$$

$$\mathcal{P} \to \mathcal{V} : T_1, T_2 \tag{49}$$

$$\mathcal{V} : x \xleftarrow{\$} \mathbb{Z}_p^\star \tag{50}$$

$$\mathcal{V} \to \mathcal{P} : x \tag{51}$$

$$\tau_x = \tau_1 \cdot x + \tau_2 \cdot x^2 + z^2 \cdot \gamma \qquad \in \mathbb{Z}_p \tag{52}$$

$$\mu = \alpha + \rho \cdot x \qquad \in \mathbb{Z}_p \tag{53}$$

$$t = \langle \mathbf{l}, \mathbf{r} \rangle \qquad \in \mathbb{Z}_p \tag{54}$$

$$\mathbf{l} = l(x) = \mathbf{a}_L - z \cdot \mathbf{1}^n + \mathbf{s}_L \cdot x \qquad \in \mathbb{Z}_p^n \tag{55}$$

$$\mathbf{r} = r(x) = \mathbf{y}^n \circ (\mathbf{a}_R + z \cdot \mathbf{1}^n + \mathbf{s}_R \cdot x) \tag{56}$$

$$+ z^2 \cdot \mathbf{2}^n \qquad \in \mathbb{Z}_p^n \tag{57}$$

$$\mathcal{P} \to \mathcal{V} : \tau_x, \mu, t, \mathbf{l}, \mathbf{r} \tag{58}$$

The verifier checks that $\mathbf{l}$ and $\mathbf{r}$ are in fact $l(x)$ and $r(x)$ and checks that $t(x) = \langle \mathbf{l}, \mathbf{r} \rangle$:

$$h'_i = h_i^{y^{-i+1}} \quad \forall i \in [1, n] \qquad \in \mathbb{G} \tag{59}$$

$$t \stackrel{?}{=} \langle \mathbf{l}, \mathbf{r} \rangle \qquad \in \mathbb{Z}_p \tag{60}$$

$$g^t h^{\tau_x} \stackrel{?}{=} g^{k(y,z) + z \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle} \cdot V^{z^2} \cdot T_1^x \cdot T_2^{x^2} \tag{61}$$

$$P = AS^x \cdot \mathbf{g}^{-z} \cdot \mathbf{h'}^{z \cdot \mathbf{y}^n + z^2 \cdot \mathbf{2}^n} \qquad \in \mathbb{G} \tag{62}$$

$$P \stackrel{?}{=} h^\mu \mathbf{g}^{\mathbf{l}} \mathbf{h'}^{\mathbf{r}} \tag{63}$$

Corollary 1 (Range Proof)**.** *The range proof presented in Section 4.1 has perfect completeness, perfect honest verifier zero-knowledge and computational special soundness.*

Proof. The range proof is a special case of the aggregated range proof from section 4.3 with $m = 1$. This is therefore a direct corollary of Theorem 3. □

4.2 Logarithmic Range Proof

Finally, we can describe the efficient range proof that uses the improved inner product argument. In the range proof protocol from Section 4.1, $\mathcal{P}$ transmits $\mathbf{l}$ and $\mathbf{r}$, which are already linear in n. We can omit this transfer by using the inner-product argument from Section 3. Note that verifying both (63) and (60)

is exactly equivalent to verifying that the witness $\mathbf{l}, \mathbf{r}$ satisfies the inner product relation (1) on public input $(\mathbf{g}, \mathbf{h}', g, P \cdot h^{-\mu}, t)$. We can therefore replace (58) with a transfer of τ_x, μ, t and an execution of an inner product argument. Instead of transmitting $\mathbf{l}$ and $\mathbf{r}$, which has a communication cost of $2 \cdot n$ elements, the inner-product argument requires transmission of just $2 \cdot \lceil \log_2(n) \rceil + 2$ elements. In total the prover sends $2 \cdot \lceil \log_2(n) \rceil + 4$ group elements and 5 elements in $\mathbb{Z}_p$.

4.3 Aggregating Logarithmic Proofs

In many of the range proof applications described in Section 1.2, a single prover needs to perform multiple range proofs at the same time.
For example, a confidential transaction often contains multiple outputs, and in fact, most transactions require a so-called *change output* to send any unspent funds back to the sender. In Provisions [DBB+15] the proof of solvency requires the exchange to conduct a range proof for every single account. Given the logarithmic size of the range proof presented in Section 4.2, there is some hope that we can perform a proof for m values which is more efficient than conducting m individual range proofs. In this section, we show that this can be achieved with a slight modification to the proof system from Section 4.1.
Concretely, we present a proof system for the following relation:

$$\{(g, h \in \mathbb{G}, \mathbf{g}, \mathbf{h} \in \mathbb{G}^{m \cdot n}, \mathbf{V} \in \mathbb{G}^m; \mathbf{v}, \boldsymbol{\gamma} \in \mathbb{Z}_p^m) : \\ V_j = h_j^{\gamma} g^{v_j} \wedge v_j \in [0, 2^n - 1] \forall j \in [1, m]\} \quad (64)$$

The prover is very similar to the prover for a simple range proof with $n \cdot m$ bits, with the following slight modifications.
In line (36), the prover should compute $\mathbf{a}_L \in \mathbb{Z}_p^{n \cdot m}$ such that $\langle \mathbf{a}_{L,[(j-1) \cdot m : j \cdot m]}, \mathbf{2}^n \rangle = v_j$ for all j in $[1, m]$, i.e. $\mathbf{a}_L$ is the concatenation of all of the bits for each v_j.
We adjust $r(X)$ accordingly so that

$$\begin{array}{l} r(X) = \mathbf{y}^{n \cdot m} \circ (\mathbf{a}_R + z \cdot \mathbf{1}^{n \cdot m} + \mathbf{s}_R \cdot X) \\ + \sum_{j=1}^{m} z^{1+j} \cdot \mathbf{0}^{(j-1) \cdot n} || \mathbf{2}^n || \mathbf{0}^{(m-j) \cdot n} \end{array} \quad (65)$$

In the computation of τ_x, we need to adjust for the randomness of each commitment V_j, so that $\tau_x = \tau_1 \cdot x + \tau_2 \cdot x^2 + \sum_{j=1}^{m} z^{1+j} \cdot \gamma_j$. Further, $k(y, z)$ is updated to incorporate more cross terms.

$$k(y, z) = -z^2 \cdot \langle \mathbf{1}^{n \cdot m}, \mathbf{y}^{n \cdot m} \rangle - \sum_{j=1}^{m} z^{j+2} \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle$$

The verification check (61) needs to be updated to include all the V_j commitments.

$$g^t h^{\tau_x} \stackrel{?}{=} g^{k(y,z) + z \cdot \langle \mathbf{1}^{n \cdot m}, \mathbf{y}^{n \cdot m} \rangle} \cdot \mathbf{V}^{z^2 \cdot \mathbf{z}^m} \cdot T_1^x \cdot T_2^{x^2}$$

Finally, we change the definition of P (62) such that it is a commitment to the new $\mathbf{r}$.

$$P = AS^x \cdot \mathbf{g}^{-z} \cdot \mathbf{h'}^{z \cdot \mathbf{y}^{n \cdot m}} \prod_{j=1}^{m} \mathbf{h'}_{[(j-1) \cdot m : j \cdot m]}^{z^{j+1} \cdot \mathbf{2}^n}$$

The aggregated range proof which makes use of the inner product argument uses $2 \cdot \lceil \log_2(n \cdot m) \rceil + 4$ group elements and 5 elements in $\mathbb{Z}_p$. Note that the proof size only grows by an additive term of $2 \cdot \log_2(m)$ when conducting multiple range proofs as opposed to a multiplicative factor of m when creating m independent range proofs.

Theorem 3. *The aggregate range proof presented in Section 4.3 has perfect completeness, perfect honest verifier zero-knowledge and computational special soundness.*

The proof for Theorem 3 is presented in Appendix B. It is analogous to the proof of Theorem 4 which is described in greater detail in Appendix C.

4.4 Non-Interactive Proof through Fiat-Shamir

For the purpose of a simpler analysis, the proof was presented as an interactive protocol with a logarithmic number of rounds. The verifier is a public coin verifier, as all the honest verifier's messages are simply random elements from $\mathbb{Z}_p^\star$. It is therefore possible to turn the protocol into a non-interactive protocol that is secure and full zero-knowledge in the random oracle model using the Fiat-Shamir heuristic [BR93]. All random challenges are replaced by hashes of the transcript up to that point. For instance $y = \mathsf{H}(A, S)$ and $z = \mathsf{H}(A, S, y)$
To avoid a trusted setup we can use such a hash function to generate $\mathbf{g}, \mathbf{h}, g, h$, i.e. the public parameters from a common random string. The hash functions needs to map from $\{0,1\}^*$ to $\mathbb{G} \backslash 1$[2]. This also makes it possible to provide a random access into the public parameters.

4.5 A Simple MPC Protocol for Bulletproofs

In several of the applications described in Section 1.2, the prover could potentially consist of multiple parties who each want to do a single range proof. For instance, multiple parties may want to create a single joined confidential transaction, where each party knows some of the inputs and outputs and needs to create range proofs for their known outputs. The joint transaction would not only be smaller than the sum of multiple transactions. It would also hide which inputs correspond to which outputs and provide some level of anonymity. These kinds of transactions are called CoinJoin transactions [Max13]. In Provisions, an exchange may distribute the private keys to multiple servers and split the customer database into separate chunks, but it still needs to produce a single short proof of solvency. Can these parties generate one Bulletproof without sharing the entire witness with each other? The parties could certainly use generic multi-party computation techniques to generate a single proof, but this might be too expensive and incur significant communication costs. This motivates the search for a simple MPC protocol specifically designed for Bulletproofs which requires little modification to the prover and is still efficient.

[2] See [BLS01] for a concrete construction of hash function into an elliptic curve

Note that for aggregate range proofs, the inputs of one range proof do not affect the output of another range proof. Given the composable structure of Bulletproofs, it turns out that m parties each having a Pedersen commitment $(V_k)_{k=1}^m$ can generate a single Bulletproof that each V_k commits to a number in some range fixed range. The protocol either uses a constant number of rounds but communication that is linear in both m and the binary encoding of the range, or it uses a logarithmic number of rounds and communication that is only linear in m. We assume for simplicity that m is a power of 2, but the protocol could be easily adapted for other m. We use the same notation as in the aggregate range proof protocol, but use k as an index to denote the kth party's message. That is $A^{(k)}$ is generated just like A but using only the inputs of party k. The MPC protocol works as follows, we assign a set of distinct generators $(\mathbf{g}^{(k)}, \mathbf{h}^{(k)})_{k=1}^m$ to each party and define $\mathbf{g}$ as the interleaved concatenation of all $\vec{g}^{(k)}$ such that $g_i = g_{\lceil \frac{i}{m} \rceil}^{((i-1) \mod m+1)}$. Define $\vec{h}$ and $\vec{h}^{(k)}$ in an analogous way. We first describe the protocol with linear communication. In each of the 3 rounds of the protocol, which correspond to the rounds of the range proof protocol, each party simply generates its part of the proof, i.e. the $A^{(k)}, S^{(k)}; T_1^{(k)}, T_2^{(k)}; \tau_x^{(k)}, \mu^{(k)}, t^{(k)}, \mathbf{l}^{(k)}, \mathbf{r}^{(k)}$ using its inputs and generators. These shares are then sent to a dealer (which could be one of the parties), who simply adds them homomorphically to generate the respective proof component, e.g. $A = \prod_{k=1}^l A^{(k)}$ and $\tau_x = \sum_{k=1}^l \tau_x^{(k)}$. In each round, the dealer generates the challenges using the Fiat-Shamir heuristic and the combined proof components and sends them to each party. Finally, 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. The dealer runs the inner product argument and generates the final proof. The protocol is complete as each proof component is simply the (homomorphic) sum of each parties' proof components, and the challenges are generated as in the original protocol. It is also secure against honest but curious adversaries as each share constitutes part of a separate zero-knowledge proof. The communication can be reduced by running a second MPC protocol for the inner product argument. The generators were selected in such a way that up to the last $\log_2(l)$ rounds each parties' witnesses are independent and the overall witness is simply the interleaved concatenation of the parties' witnesses. Therefore, parties simply compute $L^{(k)}, R^{(k)}$ in each round and a dealer computes L, R as the homomorphic sum of the shares. The dealer then again generates the challenge and sends it to each party. In the final round the parties send their witness to the dealer who finishes the protocol and constructs the final proof.

A similar protocol can be used for arithmetic circuits if the circuit is decomposable into separate independent circuits. Whether an efficient custom MPC protocol can be constructed for more complicated circuits remains an open problem.

4.6 Perfectly Binding Commitments and Proofs

Bulletproofs, like the range proofs currently used in confidential transactions, are computationally binding. An adversary that could break the discrete logarithm assumption could generate acceptable range proofs for a value outside the correct range. On the other hand, the commitments are perfectly hiding and Bulletproofs are perfect zero-knowledge, so that even an all powerful adversary cannot learn which value was committed to. Commitment schemes which are simultaneously perfectly-binding and perfectly-hiding commitments are impossible, so when designing commitment schemes and proof systems, we need to decide which properties are more important. For cryptocurrencies, the binding property is more important than the hiding property [RM]. An adversary that can break the binding property of the commitment scheme or the soundness of the proof system can generate coins out of thin air and thus create uncontrolled but undetectable inflation rendering the currency useless. Giving up the privacy of a transaction is much less harmful as the sender of the transaction or the owner of an account is harmed at worst. Unfortunately, it seems difficult to create Bulletproofs from binding commitments. The efficiency of the system relies on vector commitments which allow the commitment to a long vector in a single group element. By definition, for perfectly binding commitment schemes, the size of the commitment must be at least the size of the message and compression is thus impossible. The works [GH98, GVW02] show that in general, interactive proofs cannot have communication costs smaller than the witness size, unless some very surprising results in complexity theory hold.

While the discrete logarithm assumption is believed to hold for classical computers, it does not hold against a quantum adversary. It is especially problematic that an adversary can create a perfectly hiding UTXO at any time, planning to open to an arbitrary value later when quantum computers are available. To defend against this, we can use the technique from Ruffing and Malatova [RM] to ensure that even though the proof is only computationally binding, it is later possible to switch to a proof system that is perfectly binding and secure against quantum adversaries. In order to do this, the prover simply publishes g^γ, which turns the Pedersen commitment to v into an ElGamal commitment. Ruffing and Matalova also show that given a small message space, e.g. numbers in the range $[0, 2^n]$, it is impossible for a computationally bounded prover to construct a commitment that an unbounded adversary could open to a different message in the small message space.

Note that the commitment is now only computationally hiding, but that switching to quantum-secure range proofs is possible. Succinct quantum-secure range proofs remain an open problem, but with a slight modification, the scheme from Poelstra et al. [PBF$^+$] can achieve statistical soundness. Instead of using Pedersen commitments, we propose using ElGamal commitments in every step of the protocol. An ElGamal commitment is a Pedersen commitment with an additional commitment g^r to the randomness used. The scheme can be improved slightly if the same g^r is used in multiple range proofs. In order to retain the hiding property, a different h must be used for every proof.

5 Zero-Knowledge Proof for Arithmetic Circuits

Bootle et al. [BCC+16] present an efficient zero-knowledge argument for arbitrary arithmetic circuits using $6\log_2(n) + 13$ elements, where n is the multiplicative complexity of the circuit. We can use our improved inner product argument to get a proof of size $2\log_2(n) + 13$ elements, while simultaneously generalizing to include committed values as inputs to the arithmetic circuit. Including committed input wires is important for many applications (notably range proofs) as otherwise the circuit would need to implement a commitment algorithm. Concretely a statement about Pedersen commitments would need to implement the group exponentiation for the group that the commitment is an element of.

Following [BCC+16], we present a proof for a Hadamard-product relation. We think of a multiplication gate of fan-in 2 as having three wires; 'left' and 'right' for the input wires, and 'output' for the output wire. In the relation, $\mathbf{a}_L$ is the vector of left inputs for each multiplication gate. Similarly, $\mathbf{a}_R$ is the vector of right inputs, and $\mathbf{a}_O = \mathbf{a}_L \circ \mathbf{a}_R$ is the vector of outputs.

[BCC+16] shows how to convert an arbitrary arithmetic circuit with n multiplication gates into a relation containing a Hadamard-product as above, with an additional $Q \leqslant 2 \cdot n$ linear constraints of the form

$$\langle \mathbf{w}_{L,q}, \mathbf{a}_L \rangle + \langle \mathbf{w}_{R,q}, \mathbf{a}_R \rangle + \langle \mathbf{w}_{O,q}, \mathbf{a}_O \rangle = c_q$$

for $1 \leqslant q \leqslant Q$, with $\mathbf{w}_{L,q}, \mathbf{w}_{R,q}, \mathbf{w}_{O,q} \in \mathbb{Z}_p^n$ and $c_q \in \mathbb{Z}_p$.

We include additional commitments V_i as part of our statement, and give a protocol for a more general relation, where the linear consistency constraints include the openings v_j of the commitments V_j.

5.1 Inner-Product Proof for Arithmetic Circuits

As with the range proof we first present a linear proof system where the prover sends two vectors that have to satisfy some inner product relation. In Section 5.2 we show that the inner product relation can be replaced with an efficient inner product argument which yields short proofs for arbitrary circuits where input wires can come from Pedersen commitments. Formally we present a proof system for the following relation.

$$\begin{aligned} &\{(g, h \in \mathbb{G}, \mathbf{g}, \mathbf{h} \in \mathbb{G}^n, \mathbf{V} \in \mathbb{G}^m, \mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O \in \mathbb{Z}_p^{Q \times n}, \\ &\mathbf{W}_V \in \mathbb{Z}_p^{Q \times m}, \mathbf{c} \in \mathbb{Z}_p^Q; \mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O \in \mathbb{Z}_p^n, \mathbf{v}, \boldsymbol{\gamma} \in \mathbb{Z}_p^m) : \\ &V_j = g^{v_j} h^{\gamma_j} \; \forall j \in [1, m] \wedge \mathbf{a}_L \circ \mathbf{a}_R = \mathbf{a}_O \\ &\wedge \mathbf{W}_L \cdot \mathbf{a}_L + \mathbf{W}_R \cdot \mathbf{a}_R + \mathbf{W}_O \cdot \mathbf{a}_O = \mathbf{W}_V \cdot \mathbf{v} + \mathbf{c}\} \end{aligned} \tag{66}$$

Let $\mathbf{W}_V \in \mathbb{Z}_p^{Q \times m}$ be the weights for a commitment V_j. The presented proof system only works for relations where $\mathbf{W}_V$ is of rank m, i.e. the columns of the matrix are all linearly independent. This restriction is minor as we can construct commitments that fulfill these linearly dependent constraints as a homomorphic combination of other commitments. Consider a vector $\mathbf{w}'_V = \mathbf{a} \cdot \mathbf{W}_V \in \mathbb{Z}_p^m$ for a

vector of scalars $\mathbf{a} \in \mathbb{Z}_p^Q$ then we can construct commitment $V' = \mathbf{v}^{\mathbf{a} \cdot \mathbf{W}_V}$. Note that if the relation holds then we can conclude that $\langle \mathbf{w}_{L,j}, \mathbf{a}_L \rangle + \langle \mathbf{w}_{R,j}, \mathbf{a}_R \rangle + \langle \mathbf{w}_{O,j}, \mathbf{a}_O \rangle = \langle \mathbf{w}'_V, \mathbf{v} \rangle + \mathbf{c}$. The protocol is presented in Protocol 3. It is split into two parts. In the first part $\mathcal{P}$ commits to $l(X), r(X), t(X)$ in the second part $\mathcal{P}$ convinces $\mathcal{V}$ that the polynomials are well formed and that $\langle l(X), r(X) \rangle = t(X)$.

Theorem 4. *The proof system presented in Protocol 3 has perfect completeness, perfect honest verifier zero-knowledge and computational special soundness.*

5.2 Logarithmic-Sized Protocol

As for the range proof, we can reduce the communication cost of the protocol by using the inner product argument. Concretely transfer (75) is altered to simply τ_x, μ, t and additionally $\mathcal{P}$ and $\mathcal{V}$ engage in an inner product argument on public input $(\mathbf{g}, \mathbf{h}', g, P \cdot h^{-\mu}, t)$. Note that the statement proven is equivalent to the verification equations (84) and (81). The inner product argument has only logarithmic communication complexity and is thus highly efficient. Note that instead of transmitting $\mathbf{l}, \mathbf{r}$ the inner product argument only requires communication of $2 \cdot \lceil \log_2(2 \cdot n) \rceil$ elements instead of $2 \cdot n$. In total the prover sends $2 \cdot \lceil \log_2(n) \rceil + 9$ group elements and 6 elements in $\mathbb{Z}_p$. Using the Fiat-Shamir heuristic as in 4.4 the protocol can be turned into an efficient non interactive proof. We report implementation details and evaluations in Section 6.

Theorem 5. *The arithmetic circuit protocol using the improved inner product argument (Protocol 1) has perfect completeness, statistical zero-knowledge and computational soundness under the discrete logarithm assumption.*

Proof. Completeness follows from the completeness of the underlying protocols. Zero-knowledge follows from the fact that $\mathbf{l}$ and $\mathbf{r}$ can be efficiently simulated, and because the simulator can simply run Protocol 1 given the simulated witness $(\mathbf{l}, \mathbf{r})$. The protocol also has a knowledge-extractor, as the extractor of the range proof can be extended to extract $\mathbf{l}$ and $\mathbf{r}$ by calling the extractor of Protocol 1. The extractor uses $O(n^3)$ valid transcripts in total, which is polynomial in λ if $n = O(\lambda)$. The extractor is thus efficient and either extracts a discrete logarithm relation or a valid witness. However, if the generators $\mathbf{g}, \mathbf{h}, g, h$ are independently generated, then finding a discrete logarithm relation between them is as hard as breaking the discrete log problem. If the discrete log assumption holds in $\mathbb{G}$ then a computationally bounded $\mathcal{P}$ cannot produce discrete-logarithm relations between independent generators. The proof system is therefore computationally sound. □

6 Performance

In Table 1 we give analytical measurements for the proof size of different range proof protocols. We compare both the proof sizes for a single proof and for m

Input: $(g, h \in \mathbb{G}, \mathbf{g}, \mathbf{h} \in \mathbb{G}^n, \mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O \in \mathbb{Z}_p^{Q \times n},$
$\mathbf{W}_V \in \mathbb{Z}_p^{Q \times m}, \mathbf{c} \in \mathbb{Z}_p^Q; \mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O \in \mathbb{Z}_p^n, \boldsymbol{\gamma} \in \mathbb{Z}_p^m)$

$\mathcal{P}$'s input: $(g, h, \mathbf{g}, \mathbf{h}, \mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O, \mathbf{W}_V, \mathbf{c}; \mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O, \boldsymbol{\gamma})$

$\mathcal{V}$'s input: $(g, h, \mathbf{g}, \mathbf{h}, \mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O, \mathbf{W}_V, \mathbf{c})$

Output: $\{\mathcal{V}$ accepts,$\mathcal{V}$ rejects $\}$

$\mathcal{P}$ computes:

$\alpha, \beta, \rho \xleftarrow{\$} \mathbb{Z}_p$

$A_I = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R} \in \mathbb{G}$

$A_O = h^\beta \mathbf{g}^{\mathbf{a}_O} \in \mathbb{G}$

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

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

$\mathcal{P} \rightarrow \mathcal{V} : A_I, A_O, S$

$\mathcal{V} : y, z \xleftarrow{\$} \mathbb{Z}_p^\star$

$\mathcal{V} \rightarrow \mathcal{P} : y, z$

$\mathcal{P}$ and $\mathcal{V}$ compute:

$\mathbf{z}_{[1:]}^{Q+1} = (z, \ldots, z^Q) \in \mathbb{Z}_p^Q$

$k(y, z) = \langle \mathbf{y}^{-n} \circ (\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_R), \mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_L \rangle$

$\mathcal{P}$ computes:

$l(X) = \mathbf{a}_L \cdot X + \mathbf{a}_O \cdot X^2 + \mathbf{y}^{-n} \circ (\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_R) \cdot X$
$+ \mathbf{s}_L \cdot X^3 \in \mathbb{Z}_p^n[X]$

$r(X) = \mathbf{y}^n \circ \mathbf{a}_R \cdot X - \mathbf{y}^n + \mathbf{z}_{[1:]}^{Q+1} \cdot (\mathbf{W}_L \cdot X + \mathbf{W}_O)$
$+ \mathbf{y}^n \circ \mathbf{s}_R \cdot X^3 \in \mathbb{Z}_p^n[X]$

$t(X) = \langle l(X), r(X) \rangle = \sum_{i=1}^{6} t_i \cdot X^i \in \mathbb{Z}_p[X]$

$\mathbf{w} = \mathbf{W}_L \cdot \mathbf{a}_L + \mathbf{W}_R \cdot \mathbf{a}_R + \mathbf{W}_O \cdot \mathbf{a}_O$

$t_2 = \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n \rangle - \langle \mathbf{a}_O, \mathbf{y}^n \rangle + \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{w} \rangle + k(y, z) \in \mathbb{Z}_p$

$\tau_i \xleftarrow{\$} \mathbb{Z}_p \quad \forall i \in [1, 3, 4, 5, 6]$

$T_i = g^{t_i} h^{\tau_i} \quad \forall i \in [1, 3, 4, 5, 6]$

$\mathcal{P} \rightarrow \mathcal{V} : T_1, T_3, T_4, T_5, T_6$

Protocol 3: Part 1: Computing commitments to $l(X), r(X)$ and $t(X)$

$$\mathcal{V} : x \xleftarrow{\$} \mathbb{Z}_p^\star \tag{67}$$

$$\mathcal{V} \rightarrow \mathcal{P} : x \tag{68}$$

$\mathcal{P}$ computes: (69)

$$\mathbf{l} = l(x) \in \mathbb{Z}_p^n \tag{70}$$

$$\mathbf{r} = r(x) \in \mathbb{Z}_p^n \tag{71}$$

$$t = \langle \mathbf{l}, \mathbf{r} \rangle \in \mathbb{Z}_p \tag{72}$$

$$\tau_x = \sum_{i=1, i\neq 2}^{6} \tau_i \cdot x^i + x^2 \cdot \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{W}_V \cdot \boldsymbol{\gamma} \rangle \in \mathbb{Z}_p \tag{73}$$

$$\mu = \alpha \cdot x + \beta \cdot x^2 + \rho \cdot x^3 \in \mathbb{Z}_p \tag{74}$$

$$\mathcal{P} \rightarrow \mathcal{V} : \tau_x, \mu, t, \mathbf{l}, \mathbf{r} \tag{75}$$

$\mathcal{V}$ computes and checks: (76)

$$h_i' = h_i^{y^{-i+1}} \quad \forall i \in [1, n] \tag{77}$$

$$W_L = \mathbf{h}'^{\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_L} \tag{78}$$

$$W_R = \mathbf{g}^{\mathbf{y}^{-n} \circ (\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_R)} \tag{79}$$

$$W_O = \mathbf{h}'^{\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_O} \tag{80}$$

$$t \stackrel{?}{=} \langle \mathbf{l}, \mathbf{r} \rangle \tag{81}$$

$$g^t h^{\tau_x} \stackrel{?}{=} g^{x^2 \cdot (k(y,z) + \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{c} \rangle)} \cdot \mathbf{V}^{x^2 \cdot (\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_V)} \cdot T_1^x \cdot \prod_{i=3}^{6} T_i^{(x^i)} \tag{82}$$

$$P = A_I^x \cdot A_O^{(x^2)} \cdot \mathbf{h}'^{-\mathbf{y}^n} \cdot W_L^x \cdot W_R^x \cdot W_O \cdot S^{(x^3)} \tag{83}$$

$$P \stackrel{?}{=} h^\mu \cdot \mathbf{g}^{\mathbf{l}} \cdot \mathbf{h}'^{\mathbf{r}} \tag{84}$$

if all checks succeed: $\mathcal{V}$ accepts (85)

else: $\mathcal{V}$ rejects (86)

Protocol 3: Part 2: Polynomial identity check for $\langle l(x), r(x) \rangle = t(x)$

proofs for the range $[0, 2^n - 1]$. We compare Bulletproofs against [PBF$^+$] and a Σ-protocol range proof where the proof commits to each bit and then shows that the commitment is to 0 or 1.

Table 1: Range proof size for m proofs. $m = 1$ is the special case of a single range proof

m range proofs for range $[0, 2^n - 1]$

	# $\mathbb{G}$ elements	# $\mathbb{Z}_p$ elements
Σ Protocol [CD98]	mn	$3mn + 1$
Poelstra et al. [PBF$^+$]	$\log_3(2)\, mn$	$2\log_3(2)\, mn + 1$
Bulletproofs	$2\big(\log_2(n) + \log_2(m)\big)$	5

The table shows that Bulletproofs have a significant advantage when providing multiple range proofs at once. The proof size for the protocol presented in Section 4.3 only grows by an additive logarithmic factor when conducting m range-proofs, while all other solutions grow mulltiplicately in m.

To evaluate the performance of Bulletproofs in practice we give a reference implementation in Java 8. For the group $\mathbb{G}$ we use the elliptic curve SECP256K1[3] which has 128 bit security. In their compressed form, elliptic curve points can be stored as 32 bytes. All cryptographic operations are implemented using Bouncy Castle 1.5.7[4]. Performance could be further increased by implementing multi-exponentiation algorithms as described in [Möl01].

Table 2 shows that Bulletproofs brings a significant improvement over [PBF$^+$], even for a single range proof, and that the proof size hardly grows when multiple range proofs are constructed. We do not directly compare against the arithmetic circuit protocol by Bootle et al. [BCC$^+$16] because their scheme would require implementing the commitment function in the circuit, i.e. elliptic curve multiplication. This would not only blow up the prover and verifier time but even at an optimistic estimate of ten thousand gates[5] for such a circuit, a 52 bit range proof would exceed 3KB. To aid future use of Bulletproofs we also implemented Protocol 3 for arithmetic circuits and provide a parser for circuits in the Pinocchio [PHGR13] format. This hooks Bulletproofs up to the Pinocchio toolchain which contains a compiler from a subset of C to the circuit format. We intent to release an open source version.

[3] `http://www.secg.org/SEC2-Ver-1.0.pdf`

[4] `https://www.bouncycastle.org`

[5] An EC multiplication takes approximately 20λ arithmetic operations using projective coordinates.

Table 2: Proof size for proving that m committed values are in $[0, 2^{64} - 1]$

	$m = 1$	$m = 2$	$m = 16$
Σ Protocol [CD98]	8KB	16KB	128KB
Poelstra et al. [PBF$^+$]	4KB	8KB	61KB
Bulletproofs	672 bytes	736 bytes	928 bytes

References

[AKR$^+$13] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. Evaluating User Privacy in Bitcoin. In *Financial Cryptography*, 2013.

[And17] Oleg Andreev. Hidden in Plain Sight: Transacting Privately on a Blockchain. `blog.chain.com`, 2017.

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, pages 56–73, 2004.

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 327–357. Springer, 2016.

[BCCT12] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Innovations in Theoretical Computer Science 2012*, pages 326–349, 2012.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 111–120, 2013.

[BCG$^+$17a] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Interactive oracle proofs with constant rate and query complexity. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 40:1–40:15, 2017.

[BCG$^+$17b] Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. Cryptology

ePrint Archive, Report 2017/872, 2017. http://eprint.iacr.org/2017/872.

[BdM93] Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In *Advances in Cryptology - EUROCRYPT '93*, pages 274–285, 1993.

[BG12] Stephanie Bayer and Jens Groth. Efficient zero-knowledge argument for correctness of a shuffle. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 263–280. Springer, 2012.

[BGB17] Benedikt Bünz, Steven Goldfeder, and Joseph Bonneau. Proofs-of-delay and randomness beacons in ethereum. *IEEE SECURITY and PRIVACY ON THE BLOCKCHAIN (IEEE S&B)*, 2017.

[BGG17] Sean Bowe, Ariel Gabizon, and Matthew D. Green. A multi-party protocol for constructing the public parameters of the pinocchio zk-snark. *IACR Cryptology ePrint Archive*, 2017:602, 2017.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 514–532. Springer, 2001.

[BMC+15] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. *IEEE Symposium on Security and Privacy*, 2015.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93*, pages 62–73, 1993.

[BSBC+17] Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, et al. Computational integrity with a public random string from quasi-linear pcps. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 551–579. Springer, 2017.

[BSBTHR17] Eli Ben-Sasson, Iddo Ben-Tov, Yinon Horesh, and Michael Riabzev. Stark: Succinct transparent argument of knowledge. https://cyber.stanford.edu/sites/default/files/elibensasson.pdf, 2017.

[BSCG+13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In *CRYPTO*, 2013.

[BSCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from Bitcoin. In *IEEE Symposium on Security and Privacy*. IEEE, 2014.

[CCS08] Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. Efficient protocols for set membership and range proofs. *Advances in Cryptology-ASIACRYPT 2008*, pages 234–252, 2008.

[CD98] Ronald Cramer and Ivan Damgård. Zero-knowledge proofs for finite field arithmetic, or: Can zero-knowledge be for free? In *CRYPTO 98*, pages 424–441. Springer, 1998.

[Cha82] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, 1982.

[CHL05] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT*, 2005.

[CLS10] Rafik Chaabouni, Helger Lipmaa, and Abhi Shelat. Additive combinatorics and discrete logarithm based range protocols. In *Information Security and Privacy - 15th Australasian Conference, ACISP 2010, Sydney, Australia, July 5-7, 2010. Proceedings*, pages 336–351, 2010.

[CRR11] Ran Canetti, Ben Riva, and Guy N Rothblum. Practical delegation of computation using multiple servers. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 445–454. ACM, 2011.

[DBB+15] G Dagher, B Bünz, Joseph Bonneau, Jeremy Clark, and D Boneh. Provisions: Privacy-preserving proofs of solvency for bitcoin exchanges (full version). Technical report, IACR Cryptology ePrint Archive, 2015.

[FS01] Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In *Crypto*, volume 1, pages 368–387. Springer, 2001.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct nizks without pcps. In *Advances in Cryptology - EUROCRYPT 2013*, pages 626–645, 2013.

[GH98] Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Inf. Process. Lett.*, 67(4):205–214, 1998.

[GI08a] Jens Groth and Yuval Ishai. Sub-linear zero-knowledge argument for correctness of a shuffle. *Advances in Cryptology–EUROCRYPT 2008*, pages 379–396, 2008.

[GI08b] Jens Groth and Yuval Ishai. Sub-linear zero-knowledge argument for correctness of a shuffle. In *Advances in Cryptology - EUROCRYPT 2008*, pages 379–396, 2008.

[Gro03] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. In *Public Key Cryptography*, volume 2567, pages 145–160. Springer, 2003.

[Gro05] Jens Groth. Non-interactive zero-knowledge arguments for voting. In *International Conference on Applied Cryptography and Network Security*, pages 467–482. Springer, 2005.

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In *Advances in Cryptology - ASIACRYPT 2010*, pages 321–340, 2010.

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016*, pages 305–326, 2016.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology - EUROCRYPT 2008*, pages 415–432, 2008.

[GVW02] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1-2):1–53, 2002.

[Jed16] TE Jedusor. Mimblewimble, 2016.

[KMS+16] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 839–858. IEEE, 2016.

[KP95] Joe Kilian and Erez Petrank. An efficient non-interactive zero-knowledge proof system for NP with general assumptions. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(38), 1995.

[Lin03] Yehuda Lindell. Parallel coin-tossing and constant-round secure two-party computation. *J. Cryptology*, 16(3):143–184, 2003.

[Lip03] 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.

[Max13] Gregory Maxwell. CoinJoin: Bitcoin privacy for the real world. `bitcointalk.org`, August 2013.

[Max16] Greg Maxwell. Confidential transactions. `https://people.xiph.org/~greg/confidential_values.txt`, 2016.

[Mic94] Silvio Micali. Cs proofs. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 436–453. IEEE, 1994.

[Möl01] Bodo Möller. Algorithms for multi-exponentiation. In *Selected Areas in Cryptography*, pages 165–180. Springer, 2001.

[Mon] Monero - Private Digital Currency . `https://getmonero.org/`.

[MP15] Gregory Maxwell and Andrew Poelstra. Borromean ring signatures. `http://diyhpl.us/~bryan/papers2/bitcoin/Borromean%20ring%20signatures.pdf`, 2015.

[MPJ+13] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. A fistful of bitcoins: characterizing payments among men with no names. In *IMC*, 2013.

[MSH17] Patrick McCorry, Siamak F Shahandashti, and Feng Hao. A smart contract for boardroom voting with maximum voter privacy. *IACR Cryptology ePrint Archive*, 2017:110, 2017.

[Nak08] S Nakamoto. Bitcoin: A peer-to-peer electionic cash system. Unpublished, 2008.

[Nef01] C Andrew Neff. A verifiable secret shuffle and its application to e-voting. In *Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 116–125. ACM, 2001.

[NM+16] Shen Noether, Adam Mackenzie, et al. Ring confidential transactions. *Ledger*, 1:1–18, 2016.

[P+91] Torben P Pedersen et al. Non-interactive and information-theoretic secure verifiable secret sharing. In *Crypto*, volume 91, pages 129–140. Springer, 1991.

[PBF+] Andrew Poelstra, Adam Back, Mark Friedenbach, Gregory Maxwell, and Pieter Wuille. Confidential assets.

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 238–252. IEEE, 2013.

[PHGR16] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: nearly practical verifiable computation. *Commun. ACM*, 59(2):103–112, 2016.

[Poe] Andrew Poelstra. Mimblewimble.

[RM] Tim Ruffing and Giulio Malavolta. Switch commitments: A safety switch for confidential transactions.

[RMSK14] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. CoinShuffle: Practical decentralized coin mixing for Bitcoin. In *ESORICS*, 2014.

[San99] Tomas Sander. Efficient accumulators without trapdoor extended abstract. *Information and Communication Security*, pages 252–262, 1999.

[TR] Jason Teutsch and Christian Reitwießner. A scalable verification solution for blockchains.

[vS13] Nicolas van Saberhagen. Cryptonote v 2. 0, 2013.

[Woo14] Gavin Wood. Ethereum: A secure decentralized transaction ledger. `http://gavwood.com/paper.pdf`, 2014.

A Proof of Theorem 2

Proof. Perfect completeness follows directly because Protocol 2 converts the relation into a relation for Protocol 1. For witness extended emulation we show that there exists an efficient extractor $\mathcal{E}$. First we show how to construct an extractor $\mathcal{E}_1$ for Protocol 1 which on input $(\mathbf{g}, \mathbf{h}, u, P)$, either extracts a witness $\mathbf{a}, \mathbf{b}, c$ such that the relation holds, or discovers a non-trivial discrete logarithm relation between $\mathbf{g}, \mathbf{h}, u$. First note that the hardness of computing a discrete log relation between $\mathbf{g}', \mathbf{h}', u$ implies the hardness of computing one between $\mathbf{g}, \mathbf{h}, u$ as defined in Protocol 1. We will, therefore, use a recursive argument showing that in each step we either extract a witness or a discrete log relation. If $n = |\mathbf{g}| = 1$, then the prover reveals the witness and the relation can simply be checked directly. Now, we show for each recursive step that on input $(\mathbf{g}, \mathbf{h}, u, P)$, we can efficiently extract a witness $\mathbf{a}, \mathbf{b}$ or a non-trivial discrete logarithm relation between $\mathbf{g}, \mathbf{h}, u$. The extractor runs the prover to get L and R. Then, using 3 different challenges x_1, x_2, x_3, the extractor obtains $\mathbf{a}_{(1)}, \mathbf{b}_{(1)}, \ldots, \mathbf{a}_{(3)}, \mathbf{b}_{(3)}$, such that

$$L^{x_i^2} P R^{x_i^{-2}} = \mathbf{g}^{\mathbf{a}_{(i)}} \mathbf{h}^{\mathbf{b}_{(i)}} u^{\langle \mathbf{a}_{(i)}, \mathbf{b}_{(i)} \rangle} \quad \forall i \in [1, 3] \tag{87}$$

Using the same 3 challenge values for x, we compute η_1, η_2, η_3 such that

$$\sum_{i=1}^{3} \eta_i \cdot x^2 = 1 \wedge \sum_{i=1}^{3} \eta_i = 0 \wedge \sum_{i=1}^{3} \eta_i \cdot x_i^{-2} = 0$$

and using these η to construct linear combinations of (87) we can compute $\mathbf{a}_L, \mathbf{b}_L$ and c_L such that $L = \mathbf{g}^{\mathbf{a}_L}\mathbf{h}^{\mathbf{b}_L}u^{c_L}$. Repeating this process with different combinations, we can also compute $\mathbf{a}_P, \mathbf{a}_R, \mathbf{b}_P, \mathbf{b}_R, c_P$ and c_R such that

$$\begin{aligned} R &= \mathbf{g}^{\mathbf{a}_R}\mathbf{h}^{\mathbf{b}_R}u^{c_R} \\ P &= \mathbf{g}^{\mathbf{a}_P}\mathbf{h}^{\mathbf{b}_P}u^{c_P} \end{aligned}$$

Given the previously extracted witness $(\mathbf{a}', \mathbf{b}', c')$, and the computed representations of L, P and R for each challenge x, we get

$$\begin{aligned} L^{x^2}PR^{x^{-2}} &= \mathbf{g}^{\mathbf{a}_L\cdot x^2+\mathbf{a}_P+\mathbf{a}_R\cdot x^{-2}} \cdot \mathbf{h}^{\mathbf{b}_L\cdot x^2+\mathbf{b}_P+\mathbf{b}_R\cdot x^{-2}} \\ &\quad \cdot u^{c_L\cdot x^2+c_P+c_R\cdot x^{-2}} \\ &= (\mathbf{g}_{[:n']}^{x^{-1}} \circ \mathbf{g}_{[n':]}^{x})^{\mathbf{a}'} \cdot (\mathbf{h}_{[:n']}^{x} \circ \mathbf{h}_{[n':]}^{x^{-1}})^{\mathbf{b}'} \cdot u^{c'} \\ &= \mathbf{g}_{[:n']}^{\mathbf{a}'\cdot x^{-1}}\mathbf{g}_{[n':]}^{\mathbf{a}'\cdot x}\mathbf{h}_{[:n']}^{\mathbf{b}'\cdot x}\mathbf{g}_{[n':]}^{\mathbf{b}'\cdot x^{-1}}u^{c'} \\ \implies \mathbf{a}'\cdot x^{-1} &= \mathbf{a}_{L,[:n']}\cdot x^2 + \mathbf{a}_{P,[:n']} + \mathbf{a}_{R,[:n']}\cdot x^{-2} \\ \wedge \quad \mathbf{a}'\cdot x &= \mathbf{a}_{L,[n':]}\cdot x^2 + \mathbf{a}_{P,[n':]} + \mathbf{a}_{R,[n':]}\cdot x^{-2} \\ \wedge \quad \mathbf{b}'\cdot x &= \mathbf{b}_{L,[:n']}\cdot x^2 + \mathbf{b}_{P,[:n']} + \mathbf{b}_{R,[:n']}\cdot x^{-2} \\ \wedge \quad \mathbf{b}'\cdot x^{-1} &= \mathbf{b}_{L,[n':]}\cdot x^2 + \mathbf{b}_{P,[n':]} + \mathbf{b}_{R,[n':]}\cdot x^{-2} \\ \wedge \quad c' &= c_L\cdot x^2 + c_P + c_R\cdot x^{-2} \end{aligned}$$

If the implications do not hold, we directly obtain a non-trivial discrete logarithm relation between the generators $(g_1, \ldots, g_n, h_1, \ldots, h_n, u)$. If the implications do hold, we can deduce that the following two equalities hold.

$$\begin{aligned} &\mathbf{a}_{L,[:n']}\cdot x^3 + (\mathbf{a}_{P,[:n']} - \mathbf{a}_{L,[n':]})\cdot x \\ &+(\mathbf{a}_{R,[:n']} - \mathbf{a}_{P,[n':]})\cdot x^{-1} - \mathbf{a}_{R,[n':]}\cdot x^{-3} = 0 \end{aligned} \tag{88}$$

$$\begin{aligned} &\mathbf{b}_{L,[n':]}\cdot x^3 + (\mathbf{b}_{P,[n':]} - \mathbf{b}_{L,[:n']})\cdot x \\ &+(\mathbf{b}_{R,[n':]} - \mathbf{b}_{P,[:n']})\cdot x^{-1} - \mathbf{b}_{R,[:n']}\cdot x^{3} = 0 \end{aligned} \tag{89}$$

The equalities (88) and (89) hold for all 3 challenges x_1, x_2, x_3. They would hold for all challenges x if and only if

$$\mathbf{a}_{L,[:n']} = \mathbf{a}_{R,[n':]} = \mathbf{b}_{R,[:n']} = \mathbf{b}_{L,[n':]} = 0 \tag{90}$$

$$\wedge \quad \mathbf{a}_{L,[n':]} = \mathbf{a}_{P,[:n']} \wedge \mathbf{a}_{R,[:n']} = \mathbf{a}_{P,[n':]} \tag{91}$$

$$\wedge \quad \mathbf{b}_{L,[:n']} = \mathbf{b}_{P,[n':]} \wedge \mathbf{b}_{R,[n':]} = \mathbf{b}_{P,[:n']} \tag{92}$$

If, however, we find a value of $\mathbf{a}_L$,$\mathbf{a}_P$,$\mathbf{a}_R$,$\mathbf{b}_L$,$\mathbf{b}_P$, or $\mathbf{b}_R$ which is not of this form, we can directly compute one of the given form, using two of the three challenges and the equations (88) and (89). This however, directly results in two distinct representations of L, P or R, which yields a non-trivial discrete logarithm relation.

Finally, using the fact that $\mathbf{a}' = \mathbf{a}_{P,[:n']} \cdot x + \mathbf{a}_{P,[n':]} \cdot x^{-1}$ and $\mathbf{b}' = \mathbf{b}_{P,[:n']} \cdot x^{-1} + \mathbf{b}_{P,[n':]} \cdot x$ we see for all 3 challenges that:

$$\begin{aligned}\langle \mathbf{a}', \mathbf{b}' \rangle &= c' \\ &= c_L \cdot x^2 + c + c_R \cdot x^{-2} \\ &= \langle \mathbf{a}_{P,[:n']} \cdot x + \mathbf{a}_{P,[n':]} \cdot x^{-1}, \mathbf{b}_{P,[:n']} \cdot x^{-1} + \mathbf{b}_{P,[n':]} \cdot x \rangle \\ &= \langle \mathbf{a}_{P,[:n']}, \mathbf{b}_{P,[n':]} \rangle \cdot x^2 + \langle \mathbf{a}_{P,[:n']}, \mathbf{b}_{P,[:n']} \rangle \\ &+ \langle \mathbf{a}_{P,[n':]}, \mathbf{b}_{P,[:n']} \rangle \cdot x^{-2}\end{aligned}$$

These equalities only hold for three distinct challenges if $\langle \mathbf{a}_P, \mathbf{b}_P \rangle = c$. Therefore, the extractor either extracts discrete logarithm relations between the generators or the witness $(\mathbf{a}_C, \mathbf{b}_C)$. Using the generalized forking lemma from [BCC+16] (see Theorem 1) we can see that the extractor uses $3^{\lceil \log_2(n) \rceil} \leqslant n^2$ challenges in total and thus runs in expected polynomial time in n and λ.
We now show that using Protocol 2 we can construct an extractor $\mathcal{E}$ that extracts a valid witness for relation (2). The extractor uses the extractor $\mathcal{E}_1$ of Protocol 1. On input $(\mathbf{g}, \mathbf{h}, u, P, c)$ $\mathcal{E}$ runs the prover with on a challenge x and uses the extractor $\mathcal{E}_1$ to get witness $\mathbf{a}, \mathbf{b}$ such that: $P \cdot u^{x \cdot c} = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} u^{x \cdot \langle \mathbf{a}, \mathbf{b} \rangle}$. Forking the $\mathcal{P}$, supplying him with a challenge x' and rerunning the extractor $\mathcal{E}_1$ yields a second witness $(\mathbf{a}', \mathbf{b}')$. Again the soundness of Protocol 1 implies that $P \cdot u^{x' \cdot c} = \mathbf{g}^{\mathbf{a}'} \mathbf{h}^{\mathbf{b}'} u^{x' \cdot \langle \mathbf{a}', \mathbf{b}' \rangle}$. From the two witnesses, we can compute:

$$u^{(x-x') \cdot c} = \mathbf{g}^{\mathbf{a}-\mathbf{a}'} \mathbf{h}^{\mathbf{b}-\mathbf{b}'} u^{x \cdot \langle \mathbf{a}, \mathbf{b} \rangle - x' \cdot \langle \mathbf{a}', \mathbf{b}' \rangle}$$

Unless $\mathbf{a} = \mathbf{a}'$ and $\mathbf{b} = \mathbf{b}'$ we get a not trivial discrete log relation between $\mathbf{g}, \mathbf{h}$ and u. Otherwise we get $u^{(x-x') \cdot c} = u^{(x-x') \cdot \langle \mathbf{a}, \mathbf{b} \rangle} \implies c = \langle \mathbf{a}, \mathbf{b} \rangle$. Thus, $(\mathbf{a}, \mathbf{b})$ is a valid witness for relation (2). Since $\mathcal{E}$ forks the prover once, and uses the efficient extractor $\mathcal{E}_1$ twice, it is also efficient. This shows that the protocol has witness extended emulation. □

B Proof of Theorem 3

Proof. Perfect completeness follows from the fact that $t_0 = k(y,z) + z \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle + z^2 \cdot \langle \mathbf{z}^m, \mathbf{v} \rangle$ for all valid witnesses. To prove perfect honest-verifier zero-knowledge we construct a simulator that produces a distribution of proofs for a given statement $(g, h \in \mathbb{G}, \mathbf{g}, \mathbf{h} \in \mathbb{G}^{n \cdot m}, \mathbf{V} \in \mathbb{G}^m)$ that is indistinguishable from valid proofs produced by an honest prover interacting with an honest verifier. The simulator chooses all proof elements and challenges uniformly at random from their respective domains or computes them directly as described in the protocol. S and T_1 are computed according to the verification equations, i.e.:

$$\begin{aligned}S &= (h^{-\mu} \cdot A \cdot \mathbf{g}^{-z \cdot \mathbf{l}} \cdot \mathbf{h'}^{z \cdot \mathbf{y}^{n \cdot m} - \mathbf{r}} \prod_{j=1}^{m} \mathbf{h'}_{[(j-1) \cdot m : j \cdot m]}^{z^{j+1} \cdot \mathbf{z}^n})^{-x^{-1}} \\ T_1 &= (h^{-\tau_x} g^{k(y,z) + z \cdot \langle \mathbf{1}^{n \cdot m}, \mathbf{y}^{n \cdot m} \rangle - t} \cdot \mathbf{V}^{z^2 \cdot \mathbf{z}^m} \cdot T_2^{x^2})^{x^{-1}}\end{aligned}$$

Finally, the simulator runs the inner-product argument with the simulated witness $(\mathbf{l}, \mathbf{r})$. All elements in the proof are either independently randomly distributed or their relationship is fully defined by the verification equations. The inner product argument remains zero knowledge as we can successfully simulate the witness, thus revealing the witness or leaking information about it does not change the zero-knowledge property of the overall protocol. The simulator runs in time $O(\mathcal{V} + \mathcal{P}_{\mathsf{InnerProduct}})$ and is thus efficient.
In order to prove special soundness, we construct an extractor $\mathcal{E}$ as follows. The extractor $\mathcal{E}$ runs the prover with n different values of y, $(Q+1)$ different values of z, and 7 different values of the challenge x. This results in $14 \cdot (Q+1) \cdot n$ valid proof transcripts. The extractor $\mathcal{E}$ first runs the extractor $\mathcal{E}_{\mathsf{InnerProduct}}$ for the inner-product argument to extract a witness $\mathbf{l}, \mathbf{r}$ to the inner product argument such that $\mathbf{g}^{\mathbf{l}}\mathbf{h}^{\mathbf{r}} = P \wedge \langle \mathbf{l}, \mathbf{r} \rangle = t$. Using this witness and 3 valid transcripts with different x challenges, $\mathcal{E}$ can compute linear combinations of (63) in order to extract $\alpha, \rho, \mathbf{a}_L, \mathbf{a}_R, \mathbf{s}_L, \mathbf{s}_R$ such that $A = h^{\alpha}\mathbf{g}^{\mathbf{a}_L}\mathbf{h}^{\mathbf{a}_R}$, as well as $S = h^{\rho}\mathbf{g}^{\mathbf{s}_L}\mathbf{h}^{\mathbf{s}_R}$.
If for any other set of challenges (x, y, z) the extractor can compute a different representation of A or S, then this yields a non-trivial discrete logarithm relation between independent generators $h, \mathbf{g}, \mathbf{h}$ which contradicts the discrete logarithm assumption.
Using these representations of A and S, as well as $\mathbf{l}$ and $\mathbf{r}$, we then find that for all challenges x, y and z

$$\begin{aligned}\mathbf{l} &= \mathbf{a}_L - z \cdot \mathbf{1}^{n \cdot m} r + \mathbf{s}_L \cdot x \\ \mathbf{r} &= \mathbf{y}^{n \cdot m} \circ (\mathbf{a}_R + z \cdot \mathbf{1}^{n \cdot m} + \mathbf{s}_R \cdot x) \\ &\quad + \sum_{j=1}^{m} z^{1+j} \cdot 0^{(j-1) \cdot n} || \mathbf{2}^n || 0^{(m-j) \cdot n}\end{aligned}$$

If these equalities do not hold for all challenges and $\mathbf{l}, \mathbf{r}$ from the transcript, then we have two distinct representations of the same group element using a set of independent generators. This would be a non-trivial discrete logarithm relation.
For given values of y and z, the extractor $\mathcal{E}$ now takes 3 transcripts with different x's and uses linear combinations of equation (61) to compute $\tau_1, \tau_2, t_1, t_2, v, \gamma$ such that

$$T_1 = g^{t_1}h^{\tau_1} \wedge T_2 = g^{t_1}h^{\tau_2} \wedge g^{v}h^{\gamma} = \prod_{j=1}^{m} \mathbf{h'}_{[(j-1) \cdot m : j \cdot m]}^{z^{j+1} \cdot \mathbf{2}^n}$$

Repeating this for m different z challenges, we can compute $(v_j, \gamma_j)_{j=1}^{m}$ such that $g^{v_j}h^{\gamma_j} = V_j \ \forall j \in [1, m]$. If for any transcript $k(y, z) + z \cdot \langle \mathbf{1}^{n \cdot m}, \mathbf{y}^{n \cdot m} \rangle + \sum_{j=1}^{m} z^{j+2} \cdot \langle \mathbf{v}_j, \mathbf{2}^n \rangle + t_1 \cdot x + t_2 \cdot x^2 \neq t$ then this yields a violation of the binding property of the Pedersen commitment, i.e. a discrete log relation between g and h. If not, then for all y, z challenges and 3 distinct challenges $X = x_j, j \in [1, 3]$:

$$\sum_{i=0}^{2} t_i \cdot X - p(X) = 0$$

with $t_0 = k(y,z) + z \cdot \langle \mathbf{1}^{n \cdot m}, \mathbf{y}^{n \cdot m} \rangle + \sum_{j=1}^{m} z^{j+2} \cdot \langle \mathbf{v}_j, \mathbf{2}^n \rangle$ and
$p(X) = \sum_{i=0}^{2} p_i \cdot X^i = \langle l(X), r(X) \rangle$. Since the polynomial $t(X) - p(X)$ is of degree 2, but has at least 3 roots (each challenge x_j), it is necessarily the zero polynomial, i.e. $t(X) = \langle l(X), r(X) \rangle$.
Since this implies that $t_0 = p_0$, the following holds for all y, z challenges:

$$\begin{aligned}
&z \cdot \langle \mathbf{1}^{n \cdot m}, \mathbf{y}^{n \cdot m} \rangle + \textstyle\sum_{j=1}^{m} z^{j+2} \cdot \langle \mathbf{v}_j, \mathbf{2}^n \rangle + k(y,z) \\
&= \langle \mathbf{a}_L, \mathbf{y}^{n \cdot m} \circ \mathbf{a}_R \rangle + z \cdot \langle \mathbf{a}_L - \mathbf{a}_R, \mathbf{y}^{n \cdot m} \rangle \\
&+ \textstyle\sum_{j=1}^{m} z^{j+1} \langle \mathbf{a}_{L,[(j-1) \cdot m : j \cdot m]}, \mathbf{2}^n \rangle \\
&+ k(y,z) \in \mathbb{Z}_p
\end{aligned}$$

Using $n \cdot m$ y challenges and $m + 2$ z challenges we can infer the following.

$$\begin{aligned}
\mathbf{a}_L \circ \mathbf{a}_R &= \mathbf{0}^{n \cdot m} && \in \mathbb{Z}_p^{n \cdot m} \\
\mathbf{a}_R &= \mathbf{a}_L - \mathbf{1}^{n \cdot m} && \in \mathbb{Z}_p^{n \cdot m} \\
v_j &= \langle \mathbf{a}_{L,[(j-1) \cdot m : j \cdot m]}, \mathbf{2}^n \rangle && \in \mathbb{Z}_p \forall j \in [1, m]
\end{aligned}$$

The first two equations imply that $\mathbf{a}_L \in \{0,1\}^{n \cdot m}$. The last equation imply that $v_j \in [0, 2^{n-1}]$ for all j. Since $g^{\mathbf{v}} h^{\boldsymbol{\gamma}} = \mathbf{V}$ we have that $(\mathbf{v}, \boldsymbol{\gamma})$ is valid witness for relation (64). The extractor rewinds the prover $3 \cdot (m+1) \cdot n \cdot O(n^2)$ times. Extraction is efficient and polynomial in λ because $n, m = O(\lambda)$. □

C Proof of Theorem 4

Proof. Perfect completeness follows from the fact that

$$\begin{aligned}
t_2 = k(y,z) + \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{W}_L \cdot \mathbf{a}_L + \mathbf{W}_R \cdot \mathbf{a}_R + \mathbf{W}_O \cdot \mathbf{a}_O \rangle \\
= k(y,z) + \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{W}_V \cdot \mathbf{v} + \mathbf{c} \rangle \quad (93)
\end{aligned}$$

whenever the prover knows a witness to the relation and is honest. To prove perfect honest-verifier zero-knowledge we construct a simulator that produces a distribution of proofs for a given statement

$$\left(\begin{array}{l} g, h \in \mathbb{G}, \mathbf{g}, \mathbf{h} \in \mathbb{G}^n, \mathbf{V} \in \mathbb{G}^m, \\ (\mathbf{w}_{L,q}, \mathbf{w}_{R,q}, \mathbf{w}_{O,q})_{q=1}^{Q} \in \mathbb{Z}_p^{n \times 3}, (\mathbf{w}_{V,q})_{q=1}^{Q} \in \mathbb{Z}_p^m, \mathbf{c} \in \mathbb{Z}_p^Q \end{array} \right)$$

that is indistinguishable from valid proofs produced by an honest prover inter-

acting with an honest verifier. The simulator acts as followZ:

$$x, y, z, \mu, \tau_x \xleftarrow{\$} \mathbb{Z}_p \tag{94}$$

$$\mathbf{l}, \mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^n \tag{95}$$

$$t = \langle \mathbf{l}, \mathbf{r} \rangle \tag{96}$$

$$A_I, A_O \xleftarrow{\$} \mathbb{G} \tag{97}$$

$$S = \begin{pmatrix} A_I^x \cdot A_O^{x^2} \cdot \mathbf{g}^{-\mathbf{l}} \mathbf{h}'^{-\mathbf{y}^n - \mathbf{r} \cdot} \\ x W_L^x \cdot W_R^x \cdot h^{-\mu} \end{pmatrix}^{-x^{-3}} \tag{98}$$

$$T_3, T_4, T_5, T_6 \xleftarrow{\$} \mathbb{G} \tag{99}$$

$$T_1 = \begin{pmatrix} h^{-\tau_x} g^{x^2 \cdot (k(y,z) + \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{c} \rangle) - t} \\ \cdot \mathbf{V}^{x^2 \cdot (\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_V)} \cdot \prod_{i=3}^{6} T_i^{x^i} \end{pmatrix}^{-x^{-1}} \tag{100}$$

$$\text{Output: } (A_I, A_O, S; y, z; T_1, (T_i)_3^6; x; \tau_x, \mu, t, \mathbf{l}, \mathbf{r}) \tag{101}$$

The values $A_I, A_O, \mathbf{l}, \mathbf{r}, \mu, \tau_x$ produced by an honest prover interacting with an honest verifier are random independent elements, i.e. if $\mathbf{s}, \rho, \alpha, \tau_1, (\tau_i)_3^6, \rho$ as well as x, y, z are chosen independently and randomly. t is the inner product of $\mathbf{l}, \mathbf{r}$ as in any verifying transcript. The simulated S is fully defined by equations (84). The honestly produced T are perfectly hiding commitments and as such random group elements. Their internal relation given t and τ_x is fully defined by equation (82), which is ensured by computing T_1 accordingly. Therefore, the transcript of the proof is identically distributed to an honestly computed proof with uniformly selected challenges. The simulator runs in time $O(\mathcal{V})$ and is thus efficient.

In order to prove special soundness we construct an extractor $\mathcal{E}$ as follows. The $\mathcal{E}$ runs the prover with n different y, $(Q+1)$ different z and 7 different x challenges. This results in $14 \cdot (Q+1) \cdot n$ valid proof transcripts. $\mathcal{E}$ takes 3 valid transcripts for $x = x_1, x_2, x_3$ and fixed y and z. From the transmitted $\mathbf{l}, \mathbf{r}, t$ for each combination of challenges, $\mathcal{E}$ can compute η_1, η_2, η_3 such that

$$\sum_{i=1}^{3} \eta_i \cdot x_i = 1 \wedge \sum_{i=1}^{3} \eta_i \cdot x^2 = \sum_{i=1}^{3} \eta_i \cdot x_i^3 = 0$$

Using these η's to compute linear combinations of equation (84), $\mathcal{E}$ computes $\alpha \in \mathbb{Z}_p, \mathbf{a}_L, \mathbf{a}_R \in \mathbb{Z}_p^n$ such that $h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R} = A_I$. If for any other set of challenges (x, y, z) the extractor can compute a different $\alpha', \mathbf{a}'_L, \mathbf{a}'_R$ such that $h^{\alpha'} \mathbf{g}^{\mathbf{a}'_L} \mathbf{h}^{\mathbf{a}'_R} = A_I = h^\alpha \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R}$, then this yields a non-trivial discrete log relation between independent generators $h, \mathbf{g}, \mathbf{h}$ which contradicts the discrete log assumption. Similarly, the extractor can use the same challenges and Equation (84) to compute unique $\beta, \rho \in \mathbb{Z}_p, \mathbf{a}_{O,L}, \mathbf{a}_{O,R}, \mathbf{s}_L, \mathbf{s}_R \in \mathbb{Z}_p^n$ such that $h^\beta \mathbf{g}^{\mathbf{a}_{O,L}} \mathbf{h}^{\mathbf{a}_{O,R}} = A_O$ and $h^\rho \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R} = S$.

Using Equation (84), we can replace A_I, A_O, S with the computed representations and read $\mathbf{l}, \mathbf{r}, t$ from the transcripts. We then find that for all challenges

x, y, z:

$$\begin{aligned}
\mathbf{l} &= \mathbf{a}_L \cdot x + \mathbf{a}_{O,L} \cdot x^2 + \mathbf{y}^{-n} \circ (\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_R) \cdot X + \mathbf{s}_L \cdot x^3 \\
\mathbf{r} &= \mathbf{y}^n \circ \mathbf{a}_R \cdot x - \mathbf{y}^n + \mathbf{z}_{[1:]}^{Q+1} \cdot (\mathbf{W}_L \cdot x + \mathbf{W}_O) \\
&\quad + \mathbf{y}^n \circ \mathbf{a}_{O,R} \cdot x^2 + \mathbf{y}^n \circ \mathbf{s}_R \cdot x^3 \\
t &= \langle \mathbf{l}, \mathbf{r} \rangle
\end{aligned}$$

If these equalities do not hold for all challenges and $\mathbf{l}, \mathbf{r}$ from the transcript, then we have two distinct representations of the same group element using a set of independent generators. This would be a non-trivial discrete log relation. We now show that t_2 indeed has the form described in (93). For a given y, z the extractor takes 7 transcripts with different x's and uses linear combinations of equation (82) to compute $(\tau_i, t_i), i \in [1, 3, \ldots, 6]$ such that $T_i = g^{t_i} h^{\tau_i}$. Note that the linear combinations have to cancel out the other $T_i^{x^i}$ terms as well as $(\mathbf{v}^{\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_V})^{x^2}$. Using these (τ_i, t_i) we can compute v, γ such that $g^v h^\gamma = \mathbf{V}^{\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_V}$. Repeating this for m different z challenges, we can compute $(v_j, \gamma_j)_{j=1}^m$ using linear combinations of $g^v h^\gamma = \mathbf{V}^{\mathbf{z}_{[1:]}^{Q+1} \cdot \mathbf{W}_V}$ such that $g^{v_j} h^{\gamma_j} = V_j \forall j \in [1, m]$. This will however only succeed if the weight vectors $\mathbf{w}_{V,j}$ are linearly independent, i.e if the matrix $\mathbf{W}_V$ has rank m. This necessarily implies that $Q \geqslant m$. If for any transcript $t_1 \cdot x + \sum_{i=3}^6 t_i \cdot x^i + x^2 \cdot (\langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{W}_V \cdot \mathbf{v} + \mathbf{c} \rangle + k(y, z)) \neq t$ then this yields a violation of the binding property of the Pedersen commitment, i.e. a discrete log relation between g and h. If not, then for all y, z challenges and 7 distinct challenges $x = x_j, j \in [1, 7]$:

$$\sum_{i=1}^{6} t_i \cdot x - p(x) = 0 \tag{102}$$

with $t_2 = \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{W}_V \cdot \mathbf{v} + \mathbf{c} \rangle + k(y, z)$ and $p(x) = \sum_{i=1}^6 p_i \cdot x^i = \langle \mathbf{l}(x), \mathbf{r}(x) \rangle$. Since the polynomial $t(x) - p(x)$ is of degree 6, but has at least 7 roots (each challenge x_j), it is necessarily the zero polynomial, i.e. $t(x) = \langle \mathbf{l}(x), \mathbf{r}(x) \rangle$. Finally, we show that this equality implies that we can extract a witness $(\mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O \in \mathbb{Z}_p^n, \mathbf{v}, \boldsymbol{\gamma} \in \mathbb{Z}_p^m)$ which satisfies the relation.
The quadratic coefficient of p is:

$$\begin{aligned}
p_2 &= \langle \mathbf{a}_L, \mathbf{y}^n \circ \mathbf{a}_R \rangle - \langle \mathbf{a}_{O,L}, \mathbf{y}^n \rangle \\
&\quad + \langle \mathbf{z}_{[1:]}^{Q+1}, \mathbf{W}_L \cdot \mathbf{a}_L + \mathbf{W}_{R,q} \cdot \mathbf{a}_R + \mathbf{W}_O \cdot \mathbf{a}_{O,L} \rangle \\
&\quad + k(y, z) \in \mathbb{Z}_p
\end{aligned}$$

The polynomial equality implies that any challenge y, z, $p_2 = t_2$. Using a fixed y and $(Q+1)$ different z challenges we can infer that all coefficients of $p_2(z) - t_2(z)$ have to be zero. Using n different y challenges, i.e. $n \cdot (Q+1)$ total transcripts we can infer the following equalities:

$$\mathbf{a}_L \circ \mathbf{a}_R - \mathbf{a}_{O,L} = \mathbf{0}^n \in \mathbb{Z}_p^n \tag{103}$$

$$\mathbf{W}_L \cdot \mathbf{a}_L + \mathbf{W}_R \cdot \mathbf{a}_R + \mathbf{W}_O \cdot \mathbf{a}_{O,L} = \mathbf{W}_V \cdot \mathbf{v} + \mathbf{c} \in \mathbb{Z}_p^Q \tag{104}$$

From equation (103) we can directly infer that $\mathbf{a}_L \circ \mathbf{a}_R = \mathbf{a}_{O,L}$. Equations (104) are exactly the linear constraints on the circuit gates.
Defining $\mathbf{a}_O = \mathbf{a}_{O,L}$, we can conclude that $(\mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O, \mathbf{v}, \boldsymbol{\gamma})$ is indeed a valid witness. The extractor rewinds the prover $14 \cdot (Q+1) \cdot n$ times. Extraction is efficient and polynomial in λ because $n, Q = O(\lambda)$. □