Compressed Σ-Protocol Theory and Practical Application to Plug & Play Secure Algorithmics

Thomas Attema[1,2,3,⋆] and Ronald Cramer[1,2,⋆⋆]

[1] CWI, Cryptology Group, Amsterdam, The Netherlands
[2] Leiden University, Mathematical Institute, Leiden, The Netherlands
[3] TNO, Cyber Security and Robustness, The Hague, The Netherlands

Abstract. Σ-Protocols provide a well-understood basis for secure algorithmics. Recently, Bulletproofs (Bootle et al., EUROCRYPT 2016, and Bünz et al., S&P 2018) have been proposed as a *drop-in* replacement in case of zero-knowledge (ZK) for arithmetic circuits, achieving logarithmic communication instead of linear. Its *pivot* is an ingenious, logarithmic-size proof of knowledge BP for certain *quadratic* relations. However, reducing ZK for *general* relations to it forces a somewhat cumbersome "reinvention" of cryptographic protocol theory.

We take a rather different viewpoint and *reconcile* Bulletproofs with Σ-Protocol Theory such that (a) simpler circuit ZK is developed *within* established theory, while (b) achieving exactly the same logarithmic communication.

The natural key here is *linearization*. First, we repurpose BPs as a blackbox *compression* mechanism for standard Σ-Protocols handling ZK proofs of general *linear relations* (on compactly committed secret vectors); *our pivot*. Second, we reduce the case of general *nonlinear* relations to *blackbox* applications of our pivot via a novel variation on *arithmetic secret sharing based techniques for Σ-Protocols* (Cramer et al., ICITS 2012). Orthogonally, we enhance versatility by enabling scenarios not previously addressed, e.g., when a secret input is dispersed across several commitments. Standard implementation platforms leading to logarithmic communication follow from a Discrete-Log assumption or a generalized Strong-RSA assumption. Also, under a Knowledge-of-Exponent Assumption (KEA) communication drops to *constant*, as in ZK-SNARKS.

All in all, our theory should more generally be useful for modular ("plug & play") design of practical cryptographic protocols; this is further evidenced by our separate work (2020) on proofs of partial knowledge.

Keywords: Σ-protocols, Bulletproofs, Zero-Knowledge, Plug-and-Play, Secure Algorithmics, ZK-SNARKS, Verifiable Computation.

⋆ thomas.attema@tno.nl
⋆⋆ cramer@cwi.nl, cramer@math.leidenuniv.nl

1 Introduction

The theory of Σ-Protocols provides a well-understood basis for *plug-and-play* secure algorithmics.[4] Recently, Bulletproofs [6, 8] have been introduced as a "drop-in replacement" for Σ-Protocols in several important applications. Notably, this includes ZK for arithmetic circuits with communication $O(\log |C| \cdot \kappa)$ bits where $|C|$ is the circuit size[5] and κ is the security parameter, down from $O(|C| \cdot \kappa)$ bits. A similar result holds for range proofs.

At the heart of Bulletproofs is an interactive proof of knowledge between a Prover and Verifier showing that a Pedersen commitment to a vector of large length n satisfies a multi-variate polynomial equation of degree 2, defined with an inner product. We refer to this PoK by BP. Concretely, suppose $\mathbb{G}$ is a cyclic group of prime order q (denoted multiplicatively) supporting discrete-log-based cryptography. Suppose, furthermore, that $\mathbf{g} = (g_1, \ldots, g_n) \in \mathbb{G}^n$ and $h \in \mathbb{G}$ (each g_i as well as h generators of $\mathbb{G}$) have been set up once-and-for-all such that, for parties that may subsequently act as provers, finding nontrivial linear relations between them is computationally as hard as computing discrete logarithms in $\mathbb{G}$. For each $\mathbf{x} \in \mathbb{Z}_q^n$, define $\mathbf{g}^{\mathbf{x}} = \prod_{i=1}^n g_i^{x_i}$. A Pedersen-commitment P to a vector $\mathbf{x} \in \mathbb{Z}_q^n$ is then computed as $P = \mathbf{g}^{\mathbf{x}} \cdot h^{\rho}$ where $\rho \in \mathbb{Z}_q$ is selected uniformly at random. This commitment is information-theoretically hiding and, on account of the set-up, computationally binding. Note that it is compact in the sense that, independently of n, a commitment is a single $\mathbb{G}$-element. Suppose that n is even and write $n = 2m$. Setting $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^m$, a Bulletproof allows the prover to prove that it can open P such that the inner-product $\langle \mathbf{x}_0, \mathbf{x}_1 \rangle$ equals some value claimed by the prover.[6]

BPs stand out in that they ingeniously reduce communication to $O(\log n)$ elements from $O(n)$ via traditional methods. Although this is at the expense of introducing logarithmic number of moves (instead of constant), its public-coin nature ensures that it can be rendered non-interactive using the Fiat-Shamir heuristic [16]. However, design of BP *applications* meet with a number of *technical difficulties*. First, BPs are not zero-knowledge, and second, cryptographic protocol theory has to be "reinvented" with the quadratic constraint proved as its "pivot". This leads to practical yet rather opaque, complex protocols where applying natural plug-and-play intuition appears hard.

1.1 Summary of Our Contributions

In this work we take a different approach. We reconcile Bulletproofs with theory of Σ-Protocols such that (a) applications can follow (established) cryptographic protocol theory, thereby dispensing with the need for "reinventing" it, while

[4] Loosely speaking, we refer to modular design of "cryptographic realizations" of standard "algorithmic tasks". In other words, this entails porting algorithms for standard tasks to cryptographic scenarios, e.g., MPC and zero-knowledge.

[5] Actually, the result only depends on the number of inputs and multiplication gates.

[6] Alternatively, this inner-product value may be taken as part of the committed vector.

(b) enjoying exactly the same communication reduction. We do this by giving a precise perspective on BPs as a *significant strengthening* of the power of Σ-protocols. We believe this novel perspective is rather useful for intuitive, plug-and-play or modular design of practical secure algorithmics. Perhaps surprisingly our approach yields the same communication complexity; up to and including the constants.

We combine two essential components. First, we isolate a natural, *alternative pivot*: compact commitment with "arbitrary linear form openings". Given a Pedersen commitment to a long vector $\mathbf{x}$, consider a ZKPoK that the prover knows $\mathbf{x}$, while also revealing, for an *arbitrary, public, linear form* L, the scalar $L(\mathbf{x})$ correctly and nothing else. This has a simple Σ-Protocol. We then *compress* it by replacing the final (long) prover-message with an appropriate BP that the prover *knows it*. Indeed, the relation that this message is required to satisfy turns out amenable to deployment of a suitable BP. As a result, PoK and honest-verifier ZK are preserved, but *overall communication* drops from linear to logarithmic. In the process, we simplify known run-time analyses of knowledge extractors involved and give concrete estimates. On top of this, we introduce further necessary utility enhancements. First, without increasing overall complexity, we show, using the pivot as black-box, how to open several linear form evaluations instead of just one. Second, using this and by plug & play with our basic theory, we show how to handle the application scenario where the secret, long vector is initially "dispersed" across several commitments, by compactifying these into a single compact commitment first. This is useful in important applications. *From this point on, the only fact about the pivot that we will need is that we have access to a compact commitment scheme that allows a ZKPoK with low overall communication, showing that the prover knows the long secret committed vector and showing the correct openings of several linear evaluations on that committed vector;* the technical details do not matter anymore.

Second, the pivot's *significance* now surfaces when integrated with a novel variation on – hitherto largely overlooked – *arithmetic secret sharing based techniques for Σ-Protocols* [13], inspired by MPC. These techniques allow for *linearization* of "nonlinear relations". Mathematically, solving the linear instances first and then "linearizing" the non-linear ones is perhaps among the most natural problem solving strategies; here, this fits seamlessly with Sigma-protocol theory and our adaptation of [13]. It is in these adaptations that free choice of linear forms in the pivot is fully exploited; the maps arising from our adaptation of [13] do not form a well-structured subclass of maps. All in all, this yields *simple* logarithmic communication solutions for circuit ZK. Similarly for range proofs, which are now trivial to design. We also offer trade-offs, i.e., "square-root" complexity in constant rounds. Our results are based on either of three assumptions, the Discrete Logarithm assumption, an assumption derived from the Strong-RSA assumption, or a Knowledge-of-Exponent derived assumption.

We proceed as follows. We start by outlining our program, in nearly exclusively conceptual fashion. We believe that the fact that it is possible to do

so further underscores our main points. Later on we detail how this program deviates exactly from the paths taken in the recent literature.

1.2 A More Detailed View of Our Program

A. Our Pivotal Σ-Protocol
We isolate a basic Σ-protocol Π_0 that, given a compact commitment to a secret vector $\mathbf{x}$ of large length n, allows to *partially* open it. Concretely, given an *arbitrary, public, linear form* L, only the value $L(\mathbf{x})$ is released and nothing else. Briefly, the prover has a compact commitment P to a long secret vector $\mathbf{x}$. By a simple twist on basic Σ-protocol theory, the prover then selects a compact commitment A to a secret random vector $\mathbf{r}$. The prover sends, as first move, this commitment A *and* the values $y = L(\mathbf{x})$ and $y' = L(\mathbf{r})$. In the second move, the verifier sends a random challenge $c \in \mathbb{Z}_q$. In the third, final move, the prover then opens the commitment AP^c to a vector $\mathbf{z}$ (i.e., $\mathbf{z}$ is its committed vector; we leave the randomness underlying the commitment implicit here). Finally, the verifier checks the opening of the commitment and checks that $L(\mathbf{z}) = cy + y'$. The communication in this Σ-protocol is dominated by the *opening of* AP^c. The latter amounts to $O(n\kappa)$ bits (where κ is the security parameter), whereas the remainder of the protocol has $O(\kappa)$ bits *in total*. That said, it is an honest-verifier zero-knowledge proof of knowledge (with unconditional soundness). In addition, we describe an amortized version of this basic Σ-protocol, i.e., a Σ-protocol Π_0^{Am} that, given s compact commitments to secret vectors $\mathbf{x}_1, \ldots, \mathbf{x}_s$ and a linear form L, allows to open $L(\mathbf{x}_1), \ldots, L(\mathbf{x}_s)$ and nothing else. The communication costs of this amortized Σ-protocol are exactly $s - 1$ elements more than that of the basic Σ-protocol (i.e., the evaluations at the $s - 1$ additional input vectors).

Using the pivotal Σ-protocol as a black-box, its utility can be *enhanced*, which will be important later on. More concretely, *many linear forms* can be opened for essentially the price of a *single one*. First, by deploying a "polynomial amortization trick" (known, e.g., from MPC) we can do any number of *nullity* checks without any substantial increase in complexity. Second, building on this trick, we can extend the utility to the opening of *several* arbitrary linear forms $L_1, \ldots, L_s$ instead of a single one, at the cost of increasing the communication by exactly $s - 1$ values in $\mathbb{Z}_q$ (i.e., the evaluations of $s - 1$ additional forms). Finally, we note the entire discussion on these enhancements holds *verbatim* when we replace linear forms by *affine forms*.[7]

Note that we have identified two distinct *intractability assumptions*, each of which supports this pivot: the Discrete Logarithm assumption (as used in prior work involving Bulletproofs [6, 8]) but also one derived from the Strong-RSA assumption (as nailed down in a recent work [9] on Bulletproofs and their improved applications). The introduction focuses on the DL assumption, but the Σ-protocol for the solution derived from the Strong-RSA assumption follows similarly. Our program can be based on either platform. In addition, we show

[7] I.e., a linear form plus a constant.

how to base the program on a specific knowledge of exponent assumption. However, such assumptions are known to be unfalsifiable and, therefore, not without controversy. The details of our pivotal Σ-protocol can be found in Section 3, and the utility enhancements are described in Section 5.

B. Compressing the Pivot
We argue that protocol Π_0 can be *compressed* using the ideas underlying Bulletproofs, yielding a protocol Π_c that has the same functionality and is still an honest-verifier zero-knowledge proof of knowledge for the relation in question, but that has communication $O(\kappa \log n)$ bits *instead*, and $O(\log n)$ moves. Technically the compression degrades the soundness from unconditional to computational, and protocols with computational soundness are called arguments of knowledge. However, we will use the terms proof and argument of knowledge interchangeably. The compression techniques directly carry over to amortized Σ-protocol Π_0^{Am}. See below for variations achieving unconditional soundness.

Main compression idea. The idea is simply as follows, starting from Π_0. Suppose that P is the commitment in question. The linear forms are constants as they are part of the relation proved, so they will not be made explicit for now. Furthermore suppose that the prover has sent the message a as first move of Π_0, and that the verifier has subsequently sent challenge c as the second move. Thus, in the third –and final– move, the prover would be required to send the reply z. The verifier would, finally, apply the verification function ϕ attached to Π_0 to check that $\phi(P; a, c, z) = 1$, and accept only if this is the case. To define the compressed protocol Π_c, instead of requiring the prover to send the long vector z, a suitable adaptation of Bulletproof's PoK (BP) will be deployed to let the prover convince the verifier that it *knows* some z such that $\phi(P; a, c, z) = 1$, which is much more efficient. Note that it is immaterial that the Bulletproof part is not zero knowledge as, in Π_0, the prover would have *revealed* z anyway.

This will ensure the claimed communication reduction, i.e., $O(\kappa \log n)$ bits in $O(\log n)$ moves. We show that, as a *trade-off*, we may opt for *constant* number of rounds (instead of logarithmic) and $O(\kappa\sqrt{n})$ communication (instead of logarithmic). But of course, in non-interactive Fiat-Shamir mode (which clearly applies here), the logarithmic variant may be preferable.

Note that this compression idea equally applies to the enhancements of the basic utility as discussed above. It gives essentially the same complexities. Of course, this assumes that the number of openings of linear forms is not too large; it is not sensitive to the number of nullity checks though. The details of the compression idea can be found in Section 4.

Refined Analysis of Knowledge Extractors. In the theory of Σ-protocols [10], it is well known that *special soundness* implies knowledge soundness with knowledge error $1/q$, where q is the size of the challenge set. This result can be shown [10] by an application of Jensen's inequality to the convex function $f(X) = X(X - 1/q)$. Recently, and particularly for the above mentioned compressing techniques, natural generalizations of special soundness have become relevant. These more general notions of special soundness can again be shown to imply knowledge

soundness. However, the proof technique using Jensen's inequality is no longer directly applicable. For this reason prior works [6, 8] resort to heavy row type arguments without computing the exact knowledge error. Here, we show that an adaptation of the proof using Jensen's inequality does apply. This results in a simple proof and a refined analysis of the protocols in this paper.[8] The details of the extractor analysis can be found in the full-version of this paper [1].

Compressed Pivot with Unconditional Soundness. In addition, we show two approaches for realizing our compressed pivot with *unconditional* knowledge soundness, rather than computational. In our first approach we simply omit the step of the BP compression in which the linear-form evaluation is incorporated into the commitment, and execute that part "in the open". This works for us here since we only consider linear constraints in the compressed pivot and no quadratic ones. As a result, unconditional soundness is achieved. This approach increases the communication costs by a factor 2.

Our second approach is based on the observation that an unconditionally sound ZKPoK for opening linear forms can be based on *black-box access* to an unconditionally sound ZKPoK for just proving knowledge of an opening of a Pedersen vector commitment. The reduction uses structural information of a given linear form (i.e., it depends on the null-space and selection of a basis for it). By removing the provisions for linear forms from the compressed pivot Π_c the required black-box is realized. The details can be found in the full-version of this paper [1].

C. Compactifying a Vector of Commitments

Our compressed pivot may be summarized as compact commitments to long secret vectors that allow for very efficient partial openings, i.e., arbitrary linear forms applied to the secret committed vector. As we show later on, this is sufficient for proving any (nonlinear) relation. To make this work, all relevant prover data (secret data vector plus secret auxiliary data, such a random coins) is required to be committed to in a *single compact commitment*.

However, in many relevant practical scenarios, we must assume that the commitment to the prover's secret data vector, about which something is to be proved in zero knowledge, has already been produced *before* the zero knowledge protocol is run. In order to handle this, we require the prover to *compactify* these commitments together with the secret auxiliary data in a single commitment.

We consider two extreme scenarios: (1) the prover has a single compact commitment to the secret data vector about which some zero knowledge proof is to be conducted and (2) same, except that the prover has *individual* commitments to the coordinates of that secret data vector. For each scenario we give a conceptually clean realization by plug & play with our basic theory. We note that scenario 1 has not been addressed by previous work.

[8] These results hold for one of several possible definitions for knowledge soundness. Alternative definitions require additional effort for these techniques to work. For more details see the full-version of this paper [1].

For the first scenario the prover uses new generators to commit to the auxiliary information. Using the compressed Σ-protocol, the prover shows that this is indeed a commitment that *exclusively* involves the new generators. Prover and verifier multiply the two compact commitments to obtain a single compact commitment to all relevant data.

For the second scenario, a basic (amortized) Σ-protocol shows that the prover knows openings to all individual commitments. From this basic protocol, we define a new Σ-protocol as follows. The prover appends the first message a of the basic protocol with a compact commitment containing all relevant data *and* the randomness sampled in the first move of the basic Σ-protocol. After receiving the challenge the prover's response can now be computed as a public linear form (parameterized by the challenge c) evaluated at the vector to which the prover committed. Instead of sending this message directly, the prover and verifier run the interactive protocol to open the associated linear form on the compact vector commitment. The verifier checks that the opening of the vector commitment is also an opening of the commitment in the Σ-protocol. As a result the prover has shown that it knows openings to all the individual commitments and that these openings are contained in the compact commitment together with the auxiliary data. The details on the compactification of vector commitments can be found in Section 5.3.

D. Plug-and-Play Secure Algorithmics from Compressed Pivot
We will now explain the power of our compressed pivot. It will turn out that we only need *black-box access*. Our key point is to show how to combine this with a hitherto largely overlooked part of Σ-protocol theory, namely the work of [13] that shows how to prove *arbitrary constraints* on committed vectors by exploiting techniques from secure multi-party computation based on arithmetic secret sharing, more concretely, the ideas underlying the Commitment Multiplication Protocol from [11]. For more information, see Section 12.5.3 in [12] for a general description of efficient zero-knowledge verification of secret multiplications in terms of arbitrary (strongly-multiplicative) arithmetic secret sharing. It is this *combination* of "compact commitments with linear openings" and arithmetic secret sharing that allows for "linearizing nonlinear relations". So this explains also why our compressed pivot does not need any "direct" provision to handle nonlinearity.

We need to make some appropriate adaptations to make this work for us here. We first outline the technique from [13] and then we discuss adaptations. The work of [13] considers homomorphic commitment schemes where the secret committed to is not a vector of large length, but a *single element* of $\mathbb{Z}_q$ instead. The primary result is a Σ-protocol showing the correctness of commitments to m multiplication triples $(\alpha_i, \beta_i, \gamma_i := \alpha_i\beta_i)$, with *low amortized complexity* for large m. In other words, the protocol verifies the multiplicative relations, and the costs per triple are relatively small.

Each of the α_i's (resp., the β_i's and γ_i's) is individually committed to. Their solution employs strongly-multiplicative packed-secret sharing. For instance, consider Shamir's scheme over $\mathbb{Z}_q$, with privacy parameter $t = 1$, but

with secret-space dimension m. This uses random polynomials of degree $\leq m$, subject to the evaluations on the points $1, \ldots, m$ comprising the desired secret vector. Note that, for each sharing, a single random $\mathbb{Z}_q$-element is required (which can be taken as the evaluation at 0).

It is important to note that, given secret vector and random element, it holds by Lagrange Interpolation that, for each $c \in \mathbb{Z}_q$, the evaluation $f(c)$ of such polynomial $f(X)$ is some public $\mathbb{Z}_q$-linear combination over the coordinates of the secret vector and the random element. Namely, consider the map that takes $m+1$ arbitrary evaluations on the points $0, \ldots, m$ and that outputs the unique polynomial $f(X)$ of degree $\leq m$ interpolating them to the evaluations of $f(X)$ in all other points. A transformation matrix describing this map does not correspond to a Vandermonde-matrix, but it can be determined from it.

Now, assume that $2m < q$ (for strong-multiplicativity). The protocol goes as follows.

- The vectors of commitments to the multiplication triples are assumed to be part of the common input.
- The prover selects a random polynomial $f(X)$ that defines a packed secret sharing of the vector $(\alpha_1, \ldots, \alpha_m)$. The prover also selects a random polynomial $g(X)$ that defines a packed secret sharing of the vector $(\beta_1, \ldots, \beta_m)$. Finally, the prover computes the product polynomial $h(X) := f(X)g(X)$ of degree $\leq 2m < q$.
- The prover commits to the random $\mathbb{Z}_q$-element for the sharing based on $f(X)$, i.e., $f(0)$, and commits to the random $\mathbb{Z}_q$-element for the sharing based on $g(X)$, i.e., $g(0)$. The prover also commits the evaluations of $h(X)$ on the points $0, m+1, \ldots, 2m$.[9] Note that the "absent" evaluations at $1, \ldots, m$ comprise the γ_i's and their commitments are already assumed to be part of the common input.
- The prover sends these commitments to the verifier.
- The verifier selects a random challenge $c \in \mathbb{Z}_q$ distinct from $1, \ldots, m$ and sends it to the prover.
- By public linear combinations, both prover and verifier can compute three commitments: one to $u := f(c)$, one to $v := g(c)$ and one to $w := h(c)$. The prover opens each of these (assuming, of course, that c is in the right range). The verifier checks each of these three openings and checks whether $w = uv$. If the committed polynomials do not satisfy $f(X)g(X) = h(X)$, and under the assumption that the commitment scheme is binding, there are at most $2m$ values of c out of the $q - m$ possibilities such that the final check goes through. So a lying prover is caught with probability greater than $1 - 2m/(q-m)$. With q exponential in the security parameter and m, say, polynomial in it, this is exponentially close to 1. Honest-verifier zero-knowledge essentially follows from 1-privacy of the secret sharing scheme.

[9] By Lagrange interpolation these points, together with the γ_i's, determine $h(X)$.

Our *first observation here* is as follows. *In the above protocol, the prover may as well use our compressed pivot as a black-box.* Indeed, the entire vector

$$\mathbf{y} = (\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_m, f(0), g(0), h(0), h(1), \ldots, h(2m)) \in \mathbb{Z}_q^{4m+3}$$

of data that the prover commits to in the protocol above can be committed to in a *single* compact commitment. Note that, by definition, $\gamma_i = h(i)$ for all $1 \leq i \leq m$. Furthermore, all of the data *opened* to the verifier is some fixed linear form on the (long) secret committed vector $\mathbf{y}$. Indeed:

1. Each of the values u, v correspond to an opening of a public linear form applied to $\mathbf{y}$. The linear form is determined by some row in a transformation matrix as addressed above, under the convention that the form takes zeros on the portion of the coordinates of $\mathbf{y}$ not relevant to the computation.
2. Similarly for the value w, except that this simply corresponds to an "evaluation of a polynomial whose coefficients are defined by a part of $\mathbf{y}$". So evaluation is a public linear form as well.

Overall, we get an honest-verifier proof of knowledge for showing correctness of m secret multiplication-triples with $O(k \log m)$ bits communication in $O(\log m)$ moves (or in constant rounds but with $O(k\sqrt{m})$ bits communication).

Our *second observation here* is as follows. Suppose we have an arithmetic circuit[10] C over $\mathbb{Z}_q$ with n inputs, s outputs and m multiplication gates.[11] We can easily turn the observation above into a solution for "circuit zero-knowledge", i.e., the prover convinces the verifier that the committed vector $\mathbf{x} \in \mathbb{Z}_q^n$ satisfies some constraint captured by a given circuit C which (w.l.o.g.) returns 0. We note that [13] also gives a solution for circuit zero-knowledge. But that one does not work for us here as it gives too large complexity. So we make some changes.

By the aforementioned compactification techniques it is *sufficient* to consider the ZK scenario where the prover wants to demonstrate that C is satisfiable; this means that we may assume that the prover commits to all relevant data (inputs *and* all auxiliary data) in a *single* compact commitment. Other ZK scenarios, in which the prover has already committed to input data, are dealt with by first *compactifying* existing commitments and auxiliary information into a single compact commitment.

The protocol goes as follows. The prover first determines the computation graph implied by instantiating the circuit C with its input vector $\mathbf{x} \in \mathbb{Z}_q^n$. The m multiplication gates in C will be handled as above, i.e., via polynomials $f(X)$, $g(X)$ and $h(X)$ defining packed-secret sharings of the left inputs, the right inputs and outputs of the multiplication gates. The prover commits to each of the coordinates of $\mathbf{x}$ and to the auxiliary data $\mathsf{aux} = (f(0), g(0), h(0), h(1), \ldots, h(2m)) \in \mathbb{Z}_q^{2m+3}$ in one single compact commitment. *The length γ of the committed vector $\mathbf{y}$ thus equals $n + 2m + 3$.*

[10] Each gate of the circuit has fan-in two, but unbounded fan-out.

[11] We only count multiplication gates with *variable* inputs. Additions and multiplications by constants are implicitly handled and immaterial to the communication.

A simple fact about arithmetic circuits shows that all wire values are accessible as affine combinations of the coefficients committed to. These affine combinations are uniquely defined by the addition and scalar multiplication gates of the circuit. This explains why, *in contrast to the discussion above*, it is no longer necessary to commit explicitly to the α_i's and the β_i's as these are now implicitly committed to via said affine functions of $\mathbf{y}$. Therefore, since the values $f(0)$, $g(0)$ are still included in $\mathbf{y}$, the polynomials $f(X)$, $g(X)$ and $h(X)$ are well-defined by $\mathbf{y}$, and their evaluations are, by composition of the appropriate maps, also affine evaluations on $\mathbf{y}$.

With the above observations in hand, the protocol is reduced to opening the affine map Φ that, on input $\mathbf{y}$, outputs $(C(\mathbf{x}), f(c), g(c), h(c))$ for a challenge $c \in \mathbb{Z}_q \setminus \{1, \ldots, m\}$ sampled uniformly at random by the verifier. First, the verifier checks that $h(c) = f(c)g(c)$ which, as above, shows that the required multiplicative relations hold with high probability. Second, the verifier checks that $C(\mathbf{x}) = 0$, which shows that the circuit is satisfiable and that the prover knows a witness $\mathbf{x}$. By the amortized nullity checks (A) the costs of these openings can be amortized. As a result, circuit zero knowledge can be done $O(k \log \gamma)$ bits in $O(\log \gamma)$ moves. In particular, the communication costs are independent of the number of output vertices s. Trade-off between communication and moves applies as above. More details on circuit ZK can be found in Section 6.

E. Range Proofs

In a basic range proof a prover wishes to commit to a secret integer v and show that this integer is in a public range, say $[0, 2^{n-1}]$. From the above circuit ZK protocols, range proofs immediately follow. A prover simply considers the bit decomposition $\mathbf{b} \in \mathbb{Z}^n$ of the integer v, the length of this decomposition determines the range. Note that v can be accessed as a linear form evaluated at $\mathbf{b}$ and thereby a commitment to $\mathbf{b}$ is an implicit commitment to v. Prover and verifier run the above circuit satisfiability protocol to commit to $\mathbf{b}$ and prove that $C(\mathbf{b}) = 0$ for $C : \mathbb{Z}_q^n \to \mathbb{Z}_q^n, \quad x \mapsto x * (1 - x)$, where $*$ represents the component-wise product. The nullity check for C shows that the committed coefficients are indeed bits. The communication complexity of this range proof is $O(\kappa \log n)$ bits.

Using the techniques described in Section 5.3, this functionality can be extended to scenario where a prover has to prove that a Pedersen commitment to $v \in \mathbb{Z}_q$ is in a certain range. The details can be found in Section 7.

F. Our Program from the Strong-RSA Assumption

Thus far, we have implemented our program in the discrete log setting, starting from Pedersen commitments and their basic Σ-protocols. Besides some minor details in the compressed pivot, we show that the above discussion holds *verbatim* for a commitment scheme based on an assumption derived from the Strong-RSA assumption. More precisely, we show how the polynomial commitment scheme from a recent work [9] can be adapted to open arbitrary linear forms. Our adaptations of the linearization techniques from [13] are directly applicable to the Strong-RSA derived pivot. The details can be found in Section 7 and the full-version of this paper.

G. Our Program from the Knowledge-of-Exponent Assumption
In addition to the discrete log and strong-RSA derived assumptions, our program can also be based on an assumption derived from the Knowledge-of-Exponent Assumption (KEA). Note that KEA is unfalsifiable and its application is not completely without controversy [27, 4]. Moreover, this approach introduces a trusted set-up phase, which might be undesirable. The main benefit of the KEA based approach is that it reduces the communication complexity from logarithmic to constant, i.e., independent of the dimension of the committed vector. In Section 9 we describe the main techniques and for more details we refer to [22].

H. Proofs of Partial Knowledge from Compressed Σ-Protocol Theory
In a ZK proof of (k, n)-partial knowledge, a prover knowing witnesses for some k-subset of n given public statements can convince the verifier of this fact without revealing which k-subset. In separate work [2], we construct logarithmic size proofs of partial knowledge *for all* k, n, by adapting our compressed Σ-protocols and repurposing ideas from [14]. So far, a linear size solution is known for all k, n [14]; logarithmic size *only* for $k = 1$, i.e., *1-out-of-n proofs* [23, 5, 25]. We note that, for $k = 1$, we nearly halve the best known communication costs.

I. Our program from Lattice Assumptions
From the work of [7] we can extract an instantiation of our compressed pivot based on lattice assumptions. Based on this, our framework can therefore be instantiated from lattice assumptions. However, lattice based proofs of knowledge in general are typically subject to a so called soundness slack that is further increased by the compression in [7]. Therefore, whether or not one follows our framework, selection of *larger* implementation parameters is warranted. Further research is required to determine if and how the implementation parameters can be improved.

1.3 Comparison with Earlier Work

Traditional solutions for circuit ZK in the discrete logarithm setting have a communication complexity that is linear in the circuit size. Building on the work of Groth [20], an ingenious recursive approach achieved logarithmic communication complexity [6]. At its heart lies an earlier version of the BP protocol discussed earlier. Further improvements were introduced in [8] and later revisited in [24]. Recently, Bünz, Fisch and Szepieniec [9] show that similar results can be derived from the Strong-RSA assumption. The main merit of the Strong-RSA derived solutions is a reduction in the number of public parameters. In addition, [9] deploys proofs of exponentiation [29] to reduce the computational complexity.

A common denominator in the aforementioned works is the use of a quadratic constraint as a main pivot. In [20], a specific inner-product relation is introduced, and it is shown how basic Σ-protocols for this relation can be enhanced to achieve sub-linear communication complexity. A similar inner-product relation lies at the foundation of the logarithmic size protocols of [6], except that it also uses an earlier version of the BP idea. In [8], it is subsequently shown that a modification of the quadratic relation leads to better constants. In [24],

more general quadratic constraints were considered with a view towards reducing *computational* complexity in specific ZK scenarios. Also they strive for a more modular approach. However, this induces (minor) communication overhead in comparison to Bulletproofs [8].

Furthermore, it is worth mentioning that in [6], as an intermediate stepping stone, a polynomial commitment scheme is constructed. A polynomial commitment is a commitment to the coefficient vector of a polynomial together with the functionality of opening the evaluation at any given point. The solution derived from the Strong-RSA assumption [9] bases itself entirely on this polynomial functionality. For general relations it uses recent, but complicated, reductions [18, 26, 30]. Constructing protocols from quadratic constraints, either directly or via a polynomial commitment scheme, leads to a complex theory in which plug-and-play secure algorithmics appears hard. Significant effort is required to realize higher level applications such as circuit ZK or range proofs.

As for zero-knowledge, the work of [8] and [24] establishes this property at a higher level, and not, as do the other works, at the level of their main pivot, which leads to additional difficulties in designing ZK protocols. In fact, in [24], zero-knowledge, reduced communication and reduced computation is achieved in an integrated manner.

The most significant difference between our approach and that of the aforementioned works is our simple and direct construction of a compressed pivot to open *arbitrary* linear forms and to combine this with the simple (MPC inspired) linearization techniques from [13]. The compression is achieved by a suitable adaptation of the BP ideas [8], and the linearization techniques discard the need for a direct provision to handle nonlinearity. Moreover, plug and play design of applications according to this compressed Σ-protocol theory is just as easy as with the standard Σ-protocol theory. Despite the conceptual simplicity, the communication complexities of our approach are, even including the constants, equal to that of Bulletproofs [8].

Note that polynomial evaluation, as used in some of the other works, of course also comes down to the evaluation of a linear form, albeit a specific one. Therefore these approaches are not amenable to the linearization techniques we use. Opening *arbitrary* linear forms therefore seems to be a sweet spot in that it achieves conceptual simplicity, both in designing ZK protocols and in implementing the pivot.

2 Notation and Conventions

In this section we introduce the basic notation used in the remainder of the paper. To this end, let us consider dummy protocol Π_d.

Let $(x; w) \in R_d$, then x is called a statement and w is called a witness for x. An interactive protocol Π_d for relation R_d is a protocol that allows a prover to convince a verifier that it knows a witness w for statement x.

The protocol's public parameters are typically a set of generators $g_1, \ldots, g_n, h$ of a group $\mathbb{G}$ of prime order q. We assume that, in the setup phase, these gen-

erators are sampled uniformly at random such that the prover does not know a non-trivial discrete log relation between them. We say that the protocol is computationally knowledge sound, under the discrete logarithm assumption, if there exists an efficient extractor that either extracts a witness or finds a non-trivial discrete log relation between the public parameters $g_1, \ldots, g_n, h$.

Furthermore the protocol Π_d takes as public input x and as prover's private input w, which we write as either $\Pi_d(x; w)$ or, in the graphical protocol description, as $\textsc{Input}(x; w)$. The verifier always implicitly outputs reject or accept. Optionally, the protocol can output a public string y to both verifier and prover, and a private string w' only to the prover. In this case we write $\textsc{Output}(y; w')$. In addition to the input and output of the protocol, the prover's claim (i.e, $(x; w) \in R_d$) is made explicit in the graphical protocol description.

Finally, we write $\mathcal{L}\left(\mathbb{Z}_q^n\right) := \{(L : Z_q^n \to \mathbb{Z}_q) : L \text{ linear}\}$ for the set of linear forms on $\mathbb{Z}_q^n$.

3 The Basic Pivot

This section formally describes the Pedersen vector commitment scheme and our pivotal Σ-protocol, as discussed in Section 1.2 (A). In addition, we describe a standard amortized Σ-protocol for opening a linear form on many commitments. Compression is described in Section 4.

3.1 The Basic Σ-protocol

The primary commitment scheme under consideration in this paper is the Pedersen vector commitment scheme.

Definition 1 (Pedersen Vector Commitment [28]). *Let $\mathbb{G}$ be an Abelian group of prime order q. Pedersen vector commitments are defined by the following setup and commitment phase:*

- *Setup:* $\mathbf{g} = (g_1, \ldots, g_n) \leftarrow_R \mathbb{G}^n$, $h \leftarrow_R \mathbb{G}$.
- *Commit:* $\textsc{Com} : \mathbb{Z}_q^n \times \mathbb{Z}_q \to \mathbb{G}, \quad (\mathbf{x}, \gamma) \mapsto h^\gamma \mathbf{g}^{\mathbf{x}} := h^\gamma \prod_{i=1}^n g_i^{x_i}$.

We define $\mathbf{g}^{\mathbf{x}} := \prod_{i=1}^n g_i^{x_i}$ and $\mathbf{g}^c := (g_1^c, g_2^c, \ldots, g_n^c)$ for any $\mathbf{g} \in \mathbb{G}^n$, $\mathbf{x} \in \mathbb{Z}_q^n$ and $c \in \mathbb{Z}_q$. Moreover, the component-wise product between two vectors $\mathbf{g}, \mathbf{h} \in \mathbb{G}^n$ is written as $\mathbf{g} * \mathbf{h} = (g_1 h_1, g_2 h_2, \ldots, g_n h_n)$.

Pedersen vector commitments are perfectly hiding and computationally binding under the assumption that the prover does not know a non-trivial discrete log relation between the generators $g_1, \ldots, g_n, h$.

To open a commitment to a linear form $L : \mathbb{Z}_q^n \to \mathbb{Z}_q$ means that the prover wishes to reveal $L(\mathbf{x})$ together with a proof of validity without revealing any additional information on $\mathbf{x}$. Achieving this functionality amounts for the prover to send the value $L(\mathbf{x})$ along with a ZKPoK for the relation

$$R = \left\{ \left(P \in \mathbb{G}, L \in \mathcal{L}\left(\mathbb{Z}_q^n\right), y \in \mathbb{Z}_q; \mathbf{x} \in \mathbb{Z}_q^n, \gamma \in \mathbb{Z}_q\right) : P = \mathbf{g}^{\mathbf{x}} h^\gamma, y = L(\mathbf{x}) \right\}. \tag{1}$$

Protocol 1, denoted by Π_0, shows a basic Σ-protocol for relation R. Π_0 was informally described in Section 1.2 (A). Theorem 1 shows that Π_0 is indeed a special honest-verifier zero-knowledge (SHVZK) Proof of Knowledge (PoK). Both the communication costs from the prover $\mathcal{P}$ to the verifier $\mathcal{V}$ and vice versa are given. Note that in the non-interactive Fiat-Shamir [16] mode the communication costs from verifier to prover might be irrelevant.

Theorem 1 (Basic Pivot). *Π_0 is a 3-move protocol for relation R. It is perfectly complete, special honest-verifier zero-knowledge and unconditionally knowledge sound with knowledge error $1/q$. Moreover, the communication costs are:*

- *$\mathcal{P} \rightarrow \mathcal{V}$: 1 element of $\mathbb{G}$ and $n+2$ elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \rightarrow \mathcal{P}$: 1 element of $\mathbb{Z}_q$.*

Protocol 1 Σ-protocol Π_0 for relation R
Σ-protocol to prove correctness of a linear form evaluation.

$\textsc{Public Parameters} : \mathbf{g} \in \mathbb{G}^n, h \in \mathbb{G}$
$\textsc{Input}(P, L, y; \mathbf{x}, \gamma)$

$P = \mathbf{g}^{\mathbf{x}} h^{\gamma} \in \mathbb{G}$
$y = L(\mathbf{x}) \in \mathbb{Z}_q$

Prover		Verifier
$\mathbf{r} \leftarrow_R \mathbb{Z}_q^n, \rho \leftarrow_R \mathbb{Z}_q$		
$t = L(\mathbf{r})$		
$A = \mathbf{g}^{\mathbf{r}} h^{\rho}$	$\xrightarrow{t, A}$	
		$c \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c}$	
$\mathbf{z} = c\mathbf{x} + \mathbf{r}$		
$\phi = c\gamma + \rho$		
	$\xrightarrow{\mathbf{z}, \phi}$	
		$\mathbf{g}^{\mathbf{z}} h^{\phi} \stackrel{?}{=} AP^c$
		$L(\mathbf{z}) \stackrel{?}{=} cy + t$

3.2 Amortization over Many Commitments

A standard amortization technique for Σ-protocols allows a prover to show correctness of s evaluations of the linear form L on s committed vectors for essentially the costs of one evaluation. For details we refer to the full-version of this paper [1].

4 Compressing the Pivot

This section shows how Bulletproof techniques can be applied to compress our pivotal Σ-protocol Π_0, as mentioned in Section 1.2 (B). The key observation is that sending the final message $\widehat{\mathbf{z}} := (\mathbf{z}, \phi) \in \mathbb{Z}_q^{n+1}$ is actually a (trivial) proof of knowledge for the relation

$$R_1 = \left\{ \left(\widehat{P}, \widehat{L}, \widehat{y}; \widehat{\mathbf{z}} \right) : \widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} = \widehat{P} \wedge \widehat{y} = \widehat{L}(\widehat{\mathbf{z}}) \right\}, \tag{2}$$

where, with respect to relation R, $\widehat{\mathbf{g}} := (g_1, \ldots, g_n, h) \in \mathbb{G}^{n+1}$, $\widehat{P} := AP^c$, $\widehat{y} := cy + t$ and $\widehat{L}(\mathbf{z}, \phi) := L(\mathbf{z})$ for all $(\mathbf{z}, \phi)$. Another PoK would also suffice, in particular a PoK with a smaller communication complexity. Moreover, it is immaterial that the PoK is zero-knowledge as the original PoK clearly is not. In [6] this observation was applied to Groth's Σ-protocol [20]. The main difference is that we start with linear form relation R, whereas Groth's Σ-protocol is for a specific quadratic relation.

Let Π be a PoK for relation R_1. We call the new protocol obtained by replacing the final move of protocol Π_0 by protocol Π the *composition* and write $\Pi \diamond \Pi_0$. Since Π_0 is SHVZK it immediately follows that the composition is also SHVZK.

The essence of Bulletproofs is a PoK, denoted by BP, with logarithmic communication complexity for the following inner product relation,

$$R_{\text{bullet}} = \left\{ \left(P \in \mathbb{G}, u \in \mathbb{Z}_q; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_q^n \right) : P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} \wedge u = \langle \mathbf{a}, \mathbf{b} \rangle \right\}, \tag{3}$$

where $\mathbf{g}, \mathbf{h} \in \mathbb{G}^n$ are the public parameters. The quadratic relation R_{bullet} is quite similar to the relation R_1 and it turns out that minor adaptations of BP give a logarithmic size PoK for relation R_1. We will now describe the components of the BP protocol, while simultaneously adapting these to our relation R_1.

4.1 Reduction from Relation R_1 to Relation R_2

The first step of the BP PoK is to incorporate the linear form into the Pedersen vector commitment. For this step an additional generator $k \in \mathbb{G}$ is required such that the prover does not know a discrete log relation between the generators $g_1, \ldots, g_n, h, k$. More precisely, the problem of finding a proof for relation R_1 is reduced to the problem of finding a proof for relation

$$R_2 = \left\{ \left(Q \in \mathbb{G}, \widetilde{L} \in \mathcal{L}\left(\mathbb{Z}_q^{n+1}\right); \widehat{\mathbf{z}} \in \mathbb{Z}_q^{n+1} \right) : Q = \widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} k^{\widetilde{L}(\widehat{\mathbf{z}})} \right\}. \tag{4}$$

where, $Q := \widehat{P} k^{\widehat{y}}$ and $\widetilde{L} := c\widehat{L}$ for a random challenge $c \in \mathbb{Z}_q$ sampled by the verifier. The reduction is described in Protocol 2 and denoted by Π_1. Lemma 1 shows that Π_1 is an argument of knowledge for relation R_1.

Lemma 1. *Π_1 is a 2-move protocol for relation R_1. It is perfectly complete and computationally knowledge sound, under the discrete logarithm assumption, with knowledge error $1/q$. Moreover, the communication costs are:*

– $\mathcal{P} \to \mathcal{V}$: $n+1$ *elements of* $\mathbb{Z}_q$.
– $\mathcal{V} \to \mathcal{P}$: 1 *element of* $\mathbb{Z}_q$.

Proof. **Completeness** follows directly.

Knowledge soundness: We show that there exists an efficient algorithm χ that, on input two accepting transcripts, either extracts a witness for R_1, or finds a non-trivial discrete log relation. So let $(c_1, \widehat{\mathbf{z}}_1)$ and $(c_2, \widehat{\mathbf{z}}_2)$ be two accepting transcripts with $c_1 \neq c_2$, then $\widehat{\mathbf{g}}^{\widehat{\mathbf{z}}_1 - \widehat{\mathbf{z}}_2} k^{c_1 \widehat{L}(\widehat{\mathbf{z}}_1) - c_2 \widehat{L}(\widehat{\mathbf{z}}_2)} = k^{(c_1 - c_2)\widehat{y}}$. Hence, either we have found a non-trivial discrete log relation, or $\widehat{\mathbf{z}}_1 = \widehat{\mathbf{z}}_2$ and $c_1 \widehat{L}(\widehat{\mathbf{z}}_1) - c_2 \widehat{L}(\widehat{\mathbf{z}}_2) = (c_1 - c_2)\widehat{y}$. In the latter case, it follows that $\widehat{L}(\widehat{\mathbf{z}}_1) = \widehat{L}(\widehat{\mathbf{z}}_2) = \widehat{y}$. Moreover, from this it follows that $\widehat{\mathbf{g}}^{\widehat{\mathbf{z}}_1} k^{c_1 \widehat{L}(\widehat{\mathbf{z}}_1)} = \widehat{P} k^{c_1 \widehat{y}}$ which implies $\widehat{\mathbf{g}}^{\widehat{\mathbf{z}}_1} = \widehat{P}$.

Hence, $\widehat{\mathbf{z}}_1$ is a witness for relation R_1. From basic Σ-protocol theory the existence of an efficient extractor now follows, which proves the theorem.

Protocol 2 Argument of Knowledge Π_1 for R_1
Reduction from relation R_1 to relation R_2.

PUBLIC PARAMETERS : $\widehat{\mathbf{g}} \in \mathbb{G}^{n+1}, k \in \mathbb{G}$
INPUT$(\widehat{P}, \widehat{L}, \widehat{y}; \widehat{\mathbf{z}})$

Prover		Verifier
	$\widehat{P} = \widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} \in \mathbb{G}$	
	$\widehat{y} = \widehat{L}(\widehat{\mathbf{z}}) \in \mathbb{Z}_q$	
	$\xleftarrow{c}$	$c \leftarrow_R \mathbb{Z}_q$
	$\xrightarrow{\widehat{\mathbf{z}}}$	
		$\widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} k^{c\widehat{L}(\widehat{\mathbf{z}})} \stackrel{?}{=} \widehat{P} k^{c\widehat{y}}$

4.2 Logarithmic Size PoK for Linear Relation R_2

Next we deploy the main technique of the Bulletproof protocol to construct an efficient PoK for relation R_2. For simplicity let us assume that $n+1$ is a power of 2. If this is not the case the vector can be appended with zeros. The protocol is recursive and in each iteration the dimension of the witness is halved until its dimension equals 2. We could add one additional step to the recursion and only send the response when the dimension equals 1. This would reduce the communication costs by one field element, but it would increase the number of group elements sent by the prover by 2.

For any even dimension m and vector $\mathbf{g} \in \mathbb{G}^m$, we define $\mathbf{g}_L = (g_1, \ldots, g_{m/2})$ as its left half and $\mathbf{g}_R = (g_{m/2+1}, \ldots, g_m)$ as its right half. The same notation is used for vectors in $\mathbb{Z}_q^m$. For a linear form $L : \mathbb{Z}_q^m \to \mathbb{Z}_q$, we define

$$L_L : \mathbb{Z}_q^{m/2} \to \mathbb{Z}_q, \quad \mathbf{x} \mapsto L(\mathbf{x}, 0), \qquad L_R : \mathbb{Z}_q^{m/2} \to \mathbb{Z}_q, \quad \mathbf{x} \mapsto L(0, \mathbf{x}), \tag{5}$$

where $(\mathbf{x}, 0), (0, \mathbf{x}) \in \mathbb{Z}_q^m$ are the vectors $\mathbf{x}$ appended with $m/2$ zeros on the right and left, respectively. Recall that the component-wise product between two vectors is denoted by $*$.

The compression is described in Protocol 3 and denoted by Π_2. Theorem 2 shows that protocol Π_2 is a proof of knowledge for relation R_2. Note that, in contrast to the compression mechanism of [8], protocol Π_2 is unconditionally knowledge sound. Theorem 2 and especially the soundness error are derived from our refined extractor analysis for which we refer to the full-version of this paper [1].

Theorem 2 (Compression Mechanism). *Π_2 is a $(2\mu+1)$-move protocol for relation R_2, where $\mu = \lceil \log_2(n+1) \rceil - 1$. It is perfectly complete and unconditionally knowledge sound with knowledge error*

$$\kappa = \frac{\sum_{i=1}^{\mu} 2q^{\mu-i}(q-2)^{i-1}}{q^{\mu}} \leq \frac{2\mu}{q}. \tag{6}$$

Moreover, the communication costs are:

- *$\mathcal{P} \to \mathcal{V}$: $2\lceil \log_2(n+1) \rceil - 2$ elements of $\mathbb{G}$ and 2 elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(n+1) \rceil - 1$ elements of $\mathbb{Z}_q$.*

Proof. **Completeness** follows directly.

Knowledge soundness follows in a similar manner as it does for the amortized Σ-protocol mentioned in Section 3.2. Namely, by the same "polynomial amortization trick" the commitments A, Q, B are combined in a single commitment $Q' := AQ^cB^{c^2}$ where c is a random challenge. Informally, if a prover can open commitment Q', it follows, with high probability, that a prover can open all three commitments A, Q and B. For completeness we include the detailed proof.

We show that Π_2 is $(3, \ldots, 3)$-special sound (see the full-version of this paper [1]), i.e., that there exists an efficient algorithm χ that, on input a depth μ $(3, \ldots, 3)$-tree of accepting transcripts finds a witness for relation R_2. Knowledge soundness then follows from Lemma 3 of the full-version of this paper [1].

For simplicity we assume that we only run one of the recursive steps, i.e., we consider the 3-move variant of protocol Π_2, where the prover sends the response $\mathbf{z}'$ regardless of its dimension, and we show that this protocol is 3-special sound. From there $(3, \ldots, 3)$-special soundness follows by an inductive argument of which we omit the details.

So let us show that there exists an efficient algorithm χ that, on input 3 accepting transcripts $(A, B, c_1, \mathbf{z}_1)$, $(A, B, c_2, \mathbf{z}_2)$, $(A, B, c_3, \mathbf{z}_3)$, with $c_i \neq c_j$ for all i, j, outputs a witness for relation R_2. Given these transcripts let us define Vandermonde matrix

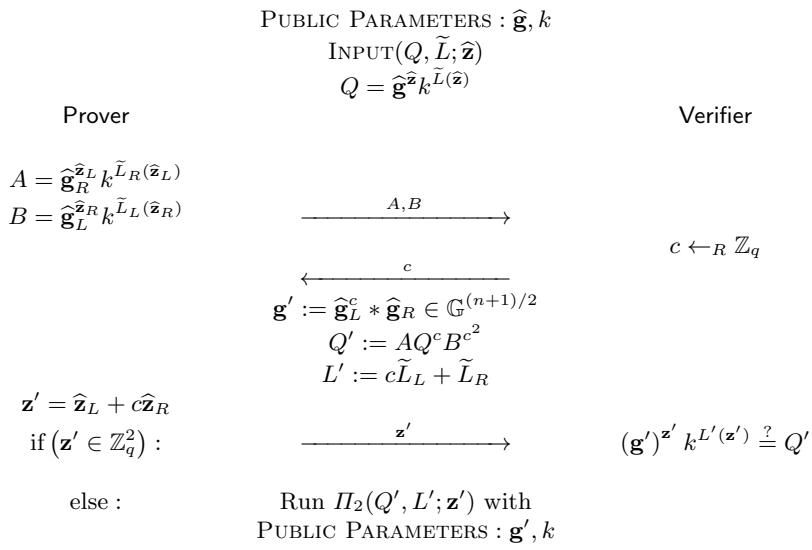
$$V = \begin{pmatrix} 1 & 1 & 1 \\ c_1 & c_2 & c_3 \\ c_1^2 & c_2^2 & c_3^2 \end{pmatrix}, \tag{7}$$

with $\det(V) = (c_3 - c_1)(c_3 - c_1)(c_3 - c_2)$. Since $c_i \neq c_j$ for all i, j, it follows that V is invertible and that we can define

$$\begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix}^T := V^{-1} \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^T . \tag{8}$$

Now it is easily seen that, for $\bar{\mathbf{z}} := \left(\sum_{i=1}^3 a_i \mathbf{z}_i, \sum_{i=1}^3 a_i c_i \mathbf{z}_i \right)$, it holds that $\mathbf{g}^{\bar{\mathbf{z}}} k^{\widetilde{L}(\bar{\mathbf{z}})} = Q$. Hence, $\mathbf{z}$ is a witness for relation R_2, which proves the claim.

Protocol 3 Compressed Proof of Knowledge Π_2 for R_2



4.3 Composing the Building Blocks

The compressed Σ-protocol Π_c for relation R is the composition of the previously mentioned protocols, i.e., $\Pi_c := \Pi_2 \diamond \Pi_1 \diamond \Pi_0$. For a graphical protocol description of Π_c we refer to the full-version of this paper [1]. Theorem 3 shows that Π_c is indeed a SHVZK argument of knowledge for relation R with a logarithmic communication complexity.

Theorem 3 (Compressed Pivot). *Π_c is a $(2\mu+3)$-move protocol for relation R, where $\mu = \lceil \log_2(n+1) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally knowledge sound, under the discrete logarithm assumption, with knowledge error*

$$\kappa = \frac{(2q-1)q^\mu + (q-1)^2 \sum_{i=1}^{\mu} 2q^{\mu-i}(q-2)^{i-1}}{q^{\mu+2}} \leq \frac{2\mu+2}{q}. \tag{9}$$

Moreover, the communication costs are:

- $\mathcal{P} \to \mathcal{V}$*:* $2\lceil \log_2(n+1) \rceil - 1$ *elements of* $\mathbb{G}$ *and* 3 *elements of* $\mathbb{Z}_q$*.*
- $\mathcal{V} \to \mathcal{P}$*:* $\lceil \log_2(n+1) \rceil + 1$ *elements of* $\mathbb{Z}_q$*.*

Proof. **Completeness** follows directly from the completeness of Π_0, Π_1 and Π_2.

SHVZK follows since Π_0 is SHVZK. The simulator for Π_c namely runs the simulator for Π_0 and continues with honest executions of Π_1 and Π_2.

Knowledge soundness follows from Lemma 3 of the full-version of this paper [1].

In a completely analogous manner, the amortized Σ-protocol Π_0^{Am} of Section 3.2 can be compressed. For the properties of the amortized and compressed Σ-protocol we refer to the full-version of this paper [1].

4.4 Compressed Pivot with Unconditional Soundness

Note that since protocol Π_1 has computational soundness so does the compressed pivot Π_c. In the full-version of this paper [1] we show two approaches for deriving an unconditionally sound compressed pivot.

4.5 A Remark on Sublinear Communication Complexity

A straightforward adaptation of the compression techniques from Section 4 allows the round complexity of the compressed pivot to be reduced from logarithmic to constant. However, this reduction comes at the cost of increasing the communication complexity from $O(\log(n))$ to $O(\sqrt{n})$ elements. For more details on this trade-off we refer to the full-version of this paper [1].

5 The Compressed Pivot as a Black-Box

From this point on, the only facts about the pivot that we need is that we have access to a compact vector commitment scheme that allows a prover to open *arbitrary* linear forms on *multiple* commitments. Hence, we assume black-box access to such a pivot. First, we treat the utility enhancements mentioned in Section 1.2 (A). Second, we describe the compactification techniques as discussed in Section 1.2 (C).

We use the following notation. We write $[\mathbf{x}]$ for a compact commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^n$, and for a (public) linear form L we write $\Pi_{\text{OPEN}}([\mathbf{x}], L; \mathbf{x})$ for the interactive protocol that reveals $L(\mathbf{x})$ and nothing else to the verifier. Recall that our notation $\Pi_{\text{OPEN}}([\mathbf{x}], L; \mathbf{x})$ means that interactive protocol Π_{OPEN} takes as public input $[\mathbf{x}]$ and L and as prover's private input $\mathbf{x}$. The communication costs of Π_{OPEN} are equal to the cost of the underlying interactive protocol (Π_c) plus 1 field element from $\mathcal{P}$ to $\mathcal{V}$ (the output of L), unless of course the output is known in advance. Similarly, we write $\Pi_{\text{OPEN}}([\mathbf{x}_1], \ldots, [\mathbf{x}_s], L; \mathbf{x}_1, \ldots, \mathbf{x}_s)$ for

the (amortized) interactive protocol that exclusively reveals $L(\mathbf{x}_i)$ for $1 \leq i \leq s$ to the verifier.

At this point, the implementation details of the compact commitment scheme do not matter anymore. However, when we give concrete knowledge errors and communication costs it is implicitly assumed that $[\cdot]$ is instantiated with Pedersen vector commitments and compressed Σ-protocol Π_c.

5.1 Many Nullity Checks for the Price of One

A "polynomial amortization trick" (known, e.g., from MPC) allows us to do many nullity checks on the committed vector $\mathbf{x}$ without a substantial increase in complexity. Consider linear forms $L_1, \ldots, L_s$ and *suppose the prover claims that* $L_i(\mathbf{x}) = 0$ *for* $i = 1 \ldots, s$. The verifier then samples $\rho \in \mathbb{Z}_q$ uniformly at random and asks the prover to open the linear form $L(\mathbf{x}) := \sum_{i=1}^{s} L_i(\mathbf{x})\rho^{i-1}$, i.e., prover and verifier run $\Pi_{\text{OPEN}}([\mathbf{x}], L; \mathbf{x})$. The opening of $L(\mathbf{x})$ equals the evaluation of some polynomial of degree at most $s-1$. If this polynomial is non-zero, it has at most $s-1$ zero's. Hence, $L(\mathbf{x}) = 0$ implies that $L_i(\mathbf{x}) = 0$ for all i with probability at least $1 - (s-1)/q$. When q is exponential and s is polynomial in the security parameter this probability is exponentially close to 1. We write $\Pi_{\text{NULLITY}}([\mathbf{x}], L_1, \ldots, L_s; , \mathbf{x})$ for this protocol. The communication costs are equal to the costs of a single nullity-check ($s = 1$) plus one additional $\mathbb{Z}_q$ element from $\mathcal{V}$ to $\mathcal{P}$ (the challenge ρ).

The above discussion holds *verbatim* when we replace the linear forms by affine forms $\Phi_1, \ldots, \Phi_s$, for which we also write $\Pi_{\text{NULLITY}}([\mathbf{x}], \Phi_1, \ldots, \Phi_s; \mathbf{x})$. Moreover, by the amortized and compressed Σ-protocol Π_c^{Am} these techniques directly carry over to the scenario where the prover makes the *same* nullity claims over many *different* commitments.

5.2 Opening Affine Maps

Many ZK scenarios can be reduced to nullity-checks and, as such, the above utility enhancement is extremely powerful. As an often encountered example, we specifically mention the functionality of opening arbitrary affine maps $\Phi : \mathbb{Z}_q^n \to \mathbb{Z}_q^s, \quad \mathbf{x} \mapsto A\mathbf{x} + b$, at the cost of increasing the communication by exactly $s-1$ values in $\mathbb{Z}_q$ in comparison to opening one linear form (i.e., the evaluations of $s-1$ additional outputs). Note that Φ is the combination of s affine forms. The protocol goes as follows. The prover reveals the evaluation $\mathbf{y} = \Phi(\mathbf{x})$ followed by an amortized nullity-check on the affine forms $\Phi_1(\mathbf{x}) - y_1, \ldots, \Phi_s(\mathbf{x}) - y_s$. For the interactive protocol that opens an affine map Φ we write $\Pi_{\text{OPEN}}([\mathbf{x}], \Phi; \mathbf{x})$.

As before, this protocol directly caries over the scenario where a prover opens the evaluations of Φ on many committed vectors. The communications costs are only increased by the additional evaluations, i.e., the communication costs of the underlying compressed Σ-protocol remain the same. Note that in this case amortization is applied twice. First, at the Σ-protocol level, allowing many commitments to be considered. Second, only requiring black-box access to the pivotal Σ-protocols, allowing many affine forms to be considered.

5.3 Compactifying a Vector of Commitments

So far, we have shown how to open many linear forms L applied to a compactly committed secret vector $\mathbf{x}$ with low complexity. Dealing with nonlinear functions of a secret-vector-of-interest $\mathbf{x}$ will, as shown in Section 6, require that the prover, at the starting point, is *also* committed to a vector aux consisting of *correlated secret randomness*. As the method will consist of opening appropriate linear forms on the *entire vector* given by the pair $(\mathbf{x}, \mathsf{aux})$, it will be assumed that the prover is committed to this pair via a *single* compact commitment.

Now, from a practical application perspective, it is likely that the prover is *already* committed to $\mathbf{x}$ before the start of a ZK proof. Consider, for example, the following two extreme cases:

- **Case 1:** The prover is committed to $\mathbf{x}$ in a *single* compact commitment. This scenario may be said to correspond to a "textbook" ZK setting.
- **Case 2:** The prover is committed to the coordinates of $\mathbf{x}$ *individually*. This scenario is relevant in practical situations with a natural dynamic where provers deliver committed data in subsequent transactions and only periodically prove in ZK some property on the compound information.

In order to deal with each of these scenarios, we need some further utility enhancements of the compressed pivot in order to bring about the desired starting point for the methods from Section 6, *without too much loss in communication*. It turns out that this is just a matter of "technology", i.e., plug and play with our compressed pivot and its basic theory suffices.

Besides these extreme cases one can consider hybrid scenarios in which the secret-vector-of-interest $\mathbf{x}$ is dispersed over various compact commitments. The methods described below *both* carry over to hybrid scenarios. The optimal approach depends on specific properties of the scenario. Namely, the communication complexity of the "Case 1 enhancement" is linear in the number of commitments, whereas the communication complexity of the "Case 2 enhancement" is linear in the (maximum) dimension of the committed vectors.

Case 1. We describe a straightforward approach. *We use the homomorphic property of Pedersen commitments.* The prover has a compact commitment P to $\mathbf{x}$. Taking from the public set-up information a new set of generators *disjoint* from the initial set that, supposedly, underlies P, the prover creates a compact commitment Q to aux. Eventually, the prover will set $P' := P \cdot Q$ as the compact commitment to the secret pair $(\mathbf{x}, \mathsf{aux})$, a *join*. But, first, the prover must show that $\mathbf{x}$ and aux "live on disjoint sets of generators". This is just a nullity check, basically. The prover shows that, in P, there is a window of zeros w.r.t. the new generators, i.e., each occurs to the power 0. Similarly for Q but with a window of zeros w.r.t. the initial set of generators. By the methods for amortized nullity checks described earlier, this is handled with logarithmic communication. In fact, for the methods of Section 6 to work, it is easy to see that it suffices to perform the check on Q only. However, since the methods of Section 6 would be applied *serially*, i.e., *after* the join above, this would incur a constant multiplicative

factor 2 loss in communication efficiency. We show how it can be done *in parallel*, thereby avoiding any such loss.

The amortized pivot allows a prover to open *one* linear form on *many* compact commitments efficiently. By the amortized nullity checks a prover can open *many* linear forms on *one* compact commitments efficiently. Together these amortization techniques almost suffice, except that they force a prover to open linear forms "intended" for one particular commitment on *other* commitments as well; they reveal the *cross-terms*. Thus, to prevent a privacy breach, we need to mask these cross-terms appropriately and we do this by constructing a small *shell* around commitments containing sufficient randomness. Masking the appropriate cross-terms returns us to the "standard" amortization scenario where the prover wishes to open one affine map on multiple compact commitments. The shells cause unintended evaluations to return random values, whereas intended evaluations are left unaltered. For the details we refer to the full-version of this paper [1].

Case 2. In this case we describe a simple, single protocol that integrates the compactification of a vector of commitments to individual coordinates of $\mathbf{x}$ together with a compact commitment to aux. See the full-version of this paper [1] for the details. Performing this integration in parallel with the methods of Section 6 is a straightforward application of the amortized nullity checks.

6 Proving Nonlinear Relations via Arithmetic Circuits

Using our compressed pivot as a black-box, this section describes how to obtain efficient zero-knowledge arguments for arbitrary arithmetic circuits. We consider arithmetic circuits C over $\mathbb{Z}_q$ with n inputs, s outputs and m multiplication gates. Addition and multiplication gates have fan-in 2 and unbounded fan-out. The number of addition gates is immaterial, as is the number of gates for scalar multiplication. For this reason m only refers to the multiplication gates that take two variable inputs. We fix an ordering $1, \ldots, n$ of the inputs and an ordering $1, \ldots, m$ of the multiplication gates.

The approach is to combine the compressed pivot with an adaptation of the work of [13] that shows how to prove arbitrary constraints on vectors of committed elements by exploiting techniques from secure multi-party computation. Concretely, we use the ideas underlying the Commitment Multiplication Protocol from [11].[12] A detailed overview of the approach has been given in Section 1.2 (D). Here, we summarize the key points and formalize the main properties of the resulting protocols.

6.1 Basic Circuit Satisfiability

First, we consider the basic circuit satisfiability scenario in which a prover shows that it knows an input $\mathbf{x} \in \mathbb{Z}_q^n$ for which the arithmetic circuit C evaluates to 0.

[12] For a general description of efficient ZK verification of secret multiplications, in terms of (strongly-multiplicative) arithmetic secret sharing, see Section 12.5.3 [12].

More precisely, we construct a ZK protocol for the following circuit satisfiability relation: $R_{cs} = \{(C; \mathbf{x}) : C(\mathbf{x}) = 0\}$.

Our approach follows the *commit and prove* paradigm, i.e., the prover commits to the witness $\mathbf{x}$ and subsequently proves that it satisfies the required relation. The terminology *circuit satisfiability* seems to suggest that we are only considering circuits for which it is hard to compute a satisfying witness $\mathbf{x}$. However, many practical scenarios consider circuits C for which it is easy to compute an $\mathbf{x}$ such that $C(\mathbf{x}) = 0$. In these scenarios the arithmetic circuit allows the prover to show that a committed vector satisfies certain properties.

If C is an affine map, i.e., without multiplication gates, the protocol follows directly from the (enhanced) functionality of our pivot. Namely, the prover commits to $\mathbf{x}$ and runs $\Pi_{\text{NULLITY}}([\mathbf{x}], C; \mathbf{x})$. Hence, addition gates and scalar multiplications, are implicitly handled since our pivot allows the opening of *arbitrary* linear forms.

Multiplication gates are handled by an appropriate adaptation of the techniques from [13]. Their primary result is a Σ-protocol showing correctness of m multiplication triples $(\alpha_i, \beta_i, \gamma_i)$. First, we recall the adaptation of their approach that uses our compressed pivot as a black-box. See also the first observation made in Section 1.2 (D). The protocol goes as follows.

- The prover selects a random polynomial $f(X) \in \mathbb{Z}_q[X]_{\leq m}$ that defines a packed secret sharing of the vector $(\alpha_1, \ldots, \alpha_m)$. The prover also selects a random polynomial $g(X) \in \mathbb{Z}_q[X]_{\leq m}$ that defines a packed secret sharing of the vector $(\beta_1, \ldots, \beta_m)$. Finally, the prover computes the product polynomial $h(X) := f(X)g(X)$ of degree $\leq 2m < q$.
- The prover commits to the vector

  $$\mathbf{y} = (\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_m, f(0), g(0), h(0), h(1), \ldots, h(2m)) \in \mathbb{Z}_q^{4m+3}$$

  in a single compact commitment and sends the commitment to the verifier. Note that, by Lagrange interpolation, the polynomials $f(X)$, $g(X)$ and $h(X)$ are uniquely defined by the vector $\mathbf{y}$.
- The verifier selects a random challenge $c \in \mathbb{Z}_q$ distinct from $1, \ldots, m$ and sends it to the prover.
- Public linear combinations of the coefficients of $\mathbf{y}$ define three values: $u := f(c)$, $v := g(c)$ and $w := h(c)$. These values are opened and the verifier checks whether $w = uv$. A cheating prover is caught with probability greater than $1 - 2m/(q - m)$ and honest-verifier zero-knowledge essentially follows from 1-privacy of the secret sharing scheme.

Now we adapt this approach to the circuit satisfiability scenario, where we let $C : \mathbb{Z}_q^n \to \mathbb{Z}_q^s$ be an arbitrary arithmetic circuits with m multiplication gates. We use a simple fact about a circuit C. Consider the computation graph induced by evaluation at *input-vector* $\mathbf{x} \in \mathbb{Z}_q^n$. Write $\gamma_1, \ldots, \gamma_m \in \mathbb{Z}_q$ for the resulting *outputs of the multiplication gates*. For each i, write $(\alpha_i, \beta_i) \in \mathbb{Z}_q^2$ for the resulting *inputs to the i-th multiplication gate*. Finally, write $\omega \in \mathbb{Z}_q^s$ for the resulting *output of the circuit*. Then, for each i, there are *affine forms*[13]

[13] $\mathbb{Z}_q$-linear forms plus a constant.

$u_i, v_i : \mathbb{Z}_q^{n+m} \to \mathbb{Z}_q$, depending only on C, such that, for all $\mathbf{x} \in \mathbb{Z}_q^n$, it holds that $\alpha_i = u_i(\mathbf{x}, \gamma_1, \ldots, \gamma_m)$ and $\beta_i = v_i(\mathbf{x}, \gamma_1, \ldots, \gamma_m)$. These forms are uniquely determined by the addition and scalar multiplication gates. Similarly, there is an affine function $w : \mathbb{Z}_q^{n+m} \to \mathbb{Z}_q^s$ such that, for all $\mathbf{x} \in \mathbb{Z}_q^n$, it holds that $\omega = w(\mathbf{x}, \gamma_1, \ldots, \gamma_m)$. In other words, a given pair $(\mathbf{x}, \gamma_1, \ldots, \gamma_m) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^m$ can be completed to an accepting computation graph if and only if $u_i(\mathbf{x}, \gamma_1, \ldots, \gamma_m) \cdot v_i(\mathbf{x}, \gamma_1, \ldots, \gamma_m) = \gamma_i$ (for $i = 1, \ldots, m$) and $w(\mathbf{x}, \gamma_1, \ldots, \gamma_m) = 0$.

The vector $\mathbf{y}$, from the above multiplication-triples approach, is now adapted as follows. The prover includes the input vector $\mathbf{x}$. *However*, the α_i's and the β_i's are *omitted* from $\mathbf{y}$. Otherwise, the vector $\mathbf{y}$ is unchanged. In particular,

$$\mathbf{y} = (\mathbf{x}, f(0), g(0), h(0), h(1), \ldots, h(2m)) \in \mathbb{Z}_q^{n+2m+3}$$

and $(\mathbf{x}, \gamma_1, \ldots, \gamma_m) := (\mathbf{x}, h(1), \ldots, h(m))$ is a subvector of $\mathbf{y}$. Subsequently, the prover compactly commits to this adapted vector $\mathbf{y}$. By the handle discussed above, the prover needs to convince the verifier that (1) $w(\mathbf{x}, \gamma_1, \ldots, \gamma_m) = 0$, and that (2) $\alpha_i \cdot \beta_i = \gamma_i$ for all $1 \leq i \leq m$. *The α_i's and β_i's are now taken as the evaluation at $(\mathbf{x}, \gamma_1, \ldots, \gamma_m)$ of the affine functions u_i, v_i introduced above.* Note that we may capture all these as affine functions *evaluated at* $\mathbf{y}$.

As for (1), checking that $w(\mathbf{x}, \gamma_1, \ldots, \gamma_m) = 0$ is just a nullity check as provided by the pivot. As for (2), the polynomials $f(X)$, $g(X)$ are *still* well-defined by the prover's compact commitment to $\mathbf{y}$. Namely, $\rho := f(0)$, i.e., the randomness underlying its selection, is *still* included in $\mathbf{y}$. As the α_i's thus defined are affine functions of $\mathbf{y}$, the prover is still (implicitly) committed to a polynomial $f(X)$ of degree $\leq m$ such that $f(0) = \rho$ and $f(i) = \alpha_i$ $(i = 1, \ldots, m)$ and evaluation of $f(X)$ in a point c is *still*, by composition of appropriate maps, an affine evaluation at $\mathbf{y}$, as enabled by the pivot. Since $\rho' := g(0)$ is also still included in $\mathbf{y}$, a similar conclusion is drawn about the β_i's, $g(X)$, and evaluation of the latter. As no changes with respect to $h(X)$ were made in $\mathbf{y}$, we conclude that the required check can be performed in the same way as before.

The costs of the different openings are reduced by applying the amortized nullity checks of Section 5.1. In fact, the communication costs are independent of the number of outputs s.

The protocol is formally described in Protocol 4 and denoted by Π_{cs}. Protocol Π_{cs} only requires black-box access to the commitment scheme $[\cdot]$. For notational convenience, we write

$$\Pi_{\text{NULLITY}}\left([\mathbf{y}], C(\mathbf{x}), f(c) - y_1, g(c) - y_2, h(c) - y_3; \mathbf{y}\right) \tag{10}$$

for the amortized nullity check on the affine forms associated to the $s + 3$ coefficients of $(C(\mathbf{x}), f(c) - z_1, g(c) - z_2, h(c) - z_3)$.

Theorem 4 shows that, when $[\cdot]$ is instantiated with Pedersen vector commitments and compressed Σ-protocol Π_c, Π_{cs} is a SHVZK argument of knowledge for relation R_{cs}. The theorem also shows that the knowledge error depends on the number of multiplication gates in the circuit. If the circuit size is polynomial in the security parameter and q is exponential, then the knowledge error is exponentially close to 0.

Theorem 4 (Basic Circuit ZK). *Π_{cs} is a $(2\mu+7)$-move protocol for the circuit relation R_{cs}, where $\mu = \lceil \log_2(n+2m+4) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally knowledge sound, under the discrete logarithm assumption, with knowledge error*

$$\kappa \leq \frac{2\mu + 2m + s + 4}{q - m}. \tag{11}$$

Moreover, the communication costs are:

- *$\mathcal{P} \to \mathcal{V}$: $2\lceil \log_2(n+2m+4) \rceil$ elements of $\mathbb{G}$ and 6 elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(n+2m+4) \rceil + 3$ elements of $\mathbb{Z}_q$.*

Proof (Sketch). **Completeness** follows directly.

Knowledge soundness: By Lagrange interpolation there exists an efficient algorithm to reconstruct a polynomial of degree t given $t+1$ evaluations. Hence, the packed secret sharing and the amortized nullity-checks are $(2m+1)$-special sound and 4-special sound, respectively. The soundness in these steps is computational, i.e., it is essential that the prover does not know a non-trivial discrete log relation. The proof now follows from Lemma 4 of the full-version of this paper [1].

SHVZK follows from 1-privacy of the secret sharing scheme and the fact that Π_c is SHVZK.

6.2 Circuit ZK from Compactification

Thus far, we have restricted ourselves to the basic circuit satisfiability scenario where the prover commits to all input and auxiliary data at once. However, there is a great variety of other scenarios, where the circuit takes as input committed values. As in Section 5.3 we consider two extreme cases for circuit ZK:

- **Case 1.** Prove that $C(\mathbf{x}) = 0$ for a vector commitment $[\mathbf{x}]$ with $\mathbf{x} \in \mathbb{Z}_q^n$.
- **Case 2.** Prove that $C(x_1, \ldots, x_n) = 0$ for commitments $[x_i]$ with $x_i \in \mathbb{Z}_q$ for all i.

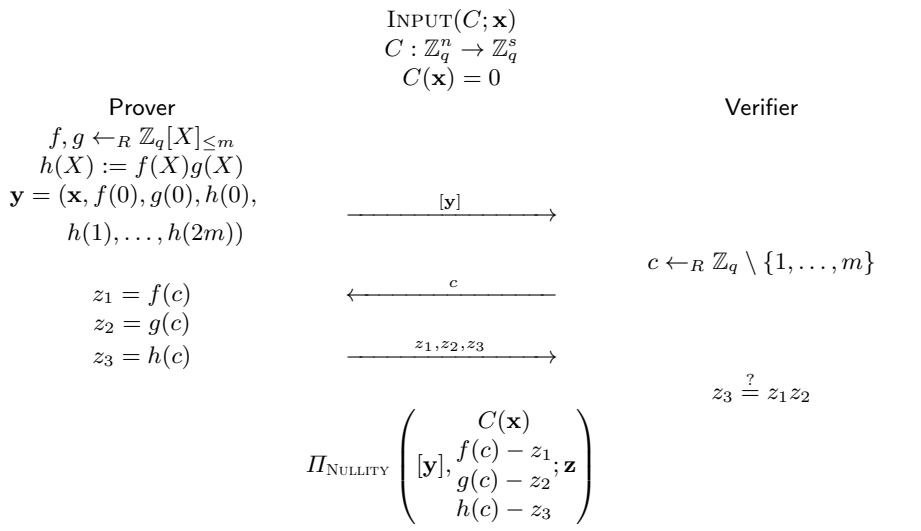
These cases are dealt with by compactifying the commitments into a single compact commitment to all relevant data. The resulting protocol for Case 1 is denoted by $\Pi_{cs}^{(1)}$ with corresponding relation $R_{cs}^{(1)}$ and its properties are given by Theorem 5. Recall that we consider arithmetic circuits C over $\mathbb{Z}_q$ with n input, s output and m multiplication gates.

Theorem 5 (Circuit ZK Case 1). *$\Pi_{cs}^{(1)}$ is a $(2\mu+9)$-move protocol for circuit relation $R_{cs}^{(1)}$, where $\mu = \lceil \log_2(n+2m+6) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally knowledge sound, under the discrete logarithm assumption, with knowledge error*

$$\kappa \leq \frac{2\mu + 2m + 4 + \max(n, s+3)}{q - m}. \tag{12}$$

Moreover, the communication costs are:

Protocol 4 Circuit Satisfiability Argument Π_{cs} for Relation R_{cs}
The polynomials f and g are sampled uniformly at random such that their evaluations in $1, \ldots, m$ coincide with the left and, respectively, right inputs of the m multiplication gates of C evaluated at $\mathbf{x}$.



– $\mathcal{P} \to \mathcal{V}$: $2\lceil \log_2(n+2m+6) \rceil + 4$ *elements of* $\mathbb{G}$ *and* 12 *elements of* $\mathbb{Z}_q$.
– $\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(n+2m+6) \rceil + 5$ *elements of* $\mathbb{Z}_q$.

The protocol for Case 2 is denoted by $\Pi_{cs}^{(2)}$ with corresponding relation $R_{cs}^{(2)}$ and its properties are given by Theorem 6. Note that in this case we can restrict ourselves to $n \leq 2m$. For if n is larger than the number of inputs to multiplication gates there must exist linear reductions that can be applied directly to the Pedersen commitments $[x_i]$ using its homomorphic properties. Therefore, the communication costs from prover to verifier are upper-bounded by $2\lceil \log_2(4m+5) \rceil + 9 \leq 2\lceil \log_2(m+2) \rceil + 13$ elements. Bulletproofs achieve a communication cost of $2\lceil \log(m) \rceil + 13$ elements. Hence, perhaps surprisingly, our plug-and-play approach almost never increases the communication costs.

Theorem 6 (Circuit ZK Case 2). *$\Pi_{cs}^{(2)}$ is a $(2\mu+7)$-move protocol for circuit relation $R_{cs}^{(2)}$, where $\mu = \lceil \log_2(n+2m+5) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally knowledge sound, under the discrete logarithm assumption, with knowledge error*

$$\kappa \leq \frac{2\mu + 2m + n + s + 5}{q - m}. \tag{13}$$

Moreover, the communication costs are:

– $\mathcal{P} \to \mathcal{V}$: $2\lceil \log_2(n+2m+5) \rceil + 1$ *elements of* $\mathbb{G}$ *and* 8 *elements of* $\mathbb{Z}_q$.
– $\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(n+2m+5) \rceil + 4$ *elements of* $\mathbb{Z}_q$.

7 Range Proofs

In a range proof a prover wishes to show that a secret committed integer v is in a public range, say $[0, 2^{n-1}]$. For our range proofs, we invoke the circuit ZK protocols of Section 6 in a black-box manner and thereby achieve a conceptual simplification of earlier solutions such as those in [6, 8]. Note that this black-box approach for range proofs can also be instantiated from the circuit ZK protocols of (e.g.) [6] and [8]. For details we refer to the full-version of this paper [1].

8 Our Program from the Strong-RSA Assumption

In this section we describe how our program can be based on Strong-RSA derived assumptions, as mentioned in Section 1.2 (F). We treat the main differences and refer to the full-version of this paper [1] and [9] for more details.

A disadvantage of the Pedersen vector commitment scheme is the number of generators required. In fact, to commit to an n-dimensional vector, $n+1$ generators of the group $\mathbb{G}$ are required. Moreover, the compressed Σ-protocol Π_c has a verification time that is linear in the dimension n.

Alternatively, vector commitment schemes can be constructed via integer commitment schemes [17, 15]. A commitment to the vector $\mathbf{x} \in \mathbb{Z}_q^n$ is then a commitment to an integer representation $\widehat{\mathbf{x}} \in \mathbb{Z}$ of $\mathbf{x}$. The integer commitment schemes of [17, 15] are constructed by using groups $\mathbb{G}$ of unknown order.

This is precisely the approach followed in a recent work of Bünz, Fisch and Szepieniec [9]. They construct a polynomial commitment scheme allowing a prover to commit to a polynomial $f \in \mathbb{Z}_q[X]$ of arbitrary degree, via a unique integer representation of its coefficient vector. A commitment to such a representation only requires two group elements $g, h \in \mathbb{G}$.

The work of [9] shows how to open arbitrary evaluations $f(a) \in \mathbb{Z}_q$ of a committed polynomial without revealing any additional information about f. Their polynomial evaluation protocol uses recursive techniques similar to those used in Bulletproofs. This approach results in a logarithmic communication complexity. In addition, [9] deploys Proofs of Exponentiation (PoE) [29] to achieve logarithmic verification time.

Their work refers to generic constructions that can be used to obtain more general ZK protocols from polynomial commitment schemes. However, we argue that these constructions are overly complicated and that a stronger functionality (vector commitment scheme with linear form openings) avoids many difficulties in the design of ZK protocols. Moreover, it turns out that the protocols of [9] only require minor adaptations to accommodate this stronger functionality. From this, an instantiation of the black-box functionality of Section 5 is derived, now based on the hardness assumptions related to the Strong-RSA assumption [3]. The techniques of Section 6 and Section 7 directly apply, and the higher level

applications inherit the logarithmic communication and computation complexity of the vector commitment scheme. The compactification methods of Section 5.3 are tailored to Pedersen (vector) commitments. Minor modifications are required to adapt these techniques to the Strong-RSA setting.

9 Our Program from the KEA

If one desires our program can also be instantiated from the the Knowledge-of-Exponent Assumption (KEA), i.e., we construct a KEA based vector commitment scheme with compact linear form openings. The techniques from Section 6 apply as before, resulting in ZK protocols for arbitrary arithmetic circuits. Basing our program on KEA reduces communication complexity from logarithmic to *constant*. The protocols do require a trusted setup that depends on the arithmetic circuit under consideration.

We stress that KEA is of a different nature than the discrete log or strong-RSA assumption. KEA is not an intractability assumption and it is unfalsifiable [27, 4]. For these reasons, its application is not completely without controversy.

We now, informally, describe the main components of the KEA based vector commitment scheme together with its ZK protocol for opening linear forms. Our approach uses the techniques of [22] and only minor adaptations are required.

A compact commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^n$ is, as before, a Pedersen vector commitment $P = h^\gamma \mathbf{g}^{\mathbf{x}}$. A ZKPoK for knowing an opening to P is another Pedersen commitment P' to $\mathbf{x}$, under the same randomness γ, using a different set of generators $h' := h^\alpha, g'_1 := g_1^\alpha, \ldots, g'_n := g_n^\alpha$. The value $\alpha \in \mathbb{Z}_q$ is sampled uniformly at random in the trusted setup phase and is only shared with a *designated* verifier. Both sets of generators are public and part of the common reference string. The proof P' is verified by checking that $P' = P^\alpha$.

The Knowledge-of-Exponent Assumption states that an adversary capable of computing pairs (P, P') with $P' = P^\alpha$, either knows α or an opening to P. From this assumption knowledge soundness follows. Correctness and zero-knowledge are immediate. Note that the resulting ZKPoK is non-interactive and its size is independent of the dimension n.

Given a bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ the verification can be done without knowledge of α, eliminating the restriction to a designated verifier. In this case verification amounts to checking that $e(P, h') = e(h, P')$.

To prove that the committed vector $\mathbf{x}$ satisfies a linear form relation $L(\mathbf{x}) = u$, the generators are taken of a specific form. More precisely, the generators are sampled under the condition that $g_i = h^{\beta^i}$, for some secret $\beta \in \mathbb{Z}_q$, for all $1 \leq i \leq n$. The associated KEA derived assumption is the n-power Knowledge-of-Exponent Assumption (n-PKEA).

Groth showed that, using this additional structure, together with the bilinear pairing, efficient circuit ZK protocols exist [22]. His protocols are easily adapted to our situation, where we simply wish to prove correctness of a linear form evaluation. The adaptation relies on the following observation. Suppose that

$\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_q^n$ is such that $L(\mathbf{z}) = \langle \mathbf{a}, \mathbf{z} \rangle$ for all $\mathbf{z} \in \mathbb{Z}_q^n$, and let us define the following polynomials: $f(Y) := \gamma + \sum_{i=1}^{n} x_i Y^i$, $g(Y) := \sum_{i=0}^{n-1} a_{n-i} Y^i$ and $h(Y) := f(Y)g(Y) = \sum_{i=0}^{2n-1} c_i Y^i$. The n-th coefficient of $h(Y)$ equals $c_n = \langle \mathbf{x}, \mathbf{a} \rangle = L(\mathbf{x})$. This observation allows for a straightforward adaptation of the product argument in [22, Section 6] , resulting in a constant size ZKPoK for the correctness of a linear form evaluation. We omit further details and refer the reader to [22].

For circuit ZK protocols we apply the techniques from Section 6 to linearize the non-linearities in a black-box manner. In contrast, other KEA based approaches use a protocol for proving quadratic relations as their main pivot and translate arithmetic circuit relations to so called quadratic span programs or QSPs [19, 21]. This translation, also called arithmetization, is not required when applying our linearization techniques. However, in contrast to other KEA based protocols, the linearization techniques render our solution interactive (although in a setting where Fiat-Shamir applies). Additionally, we note that this approach achieves constant verification complexity, in contrast to the linear complexity of the DL based approach, i.e., our KEA based protocol is a ZK-SNARK.

10 Acknowledgements

We would like to thank Serge Fehr, Toon Segers and Thijs Veugen for their careful reading of the previous version and for their helpful comments. We would also like to thank Jens Groth for useful editorial comments and pointers. Thomas Attema has been supported by EU H2020 project No 780701 (PROMETHEUS). Ronald Cramer has been supported by ERC ADG project No 74079 (ALGSTRONGCRYPTO) and by the NWO Gravitation Programme (QSC).

References

1. Full-version of this paper. IACR ePrint 2020/152
2. Attema, T., Cramer, R., Fehr, S.: Compressing proofs of k-out-of-n partial knowledge. IACR ePrint 2020/753
3. Baric, N., Pfitzmann, B.: Collision-free accumulators and fail-stop signature schemes without trees. In: EUROCRYPT 1997. pp. 480–494
4. Bitansky, N., Canetti, R., Paneth, O., Rosen, A.: On the existence of extractable one-way functions. In: STOC ACM 2014. pp. 505–514
5. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short accountable ring signatures based on DDH. In: ESORICS 2015. pp. I: 243–265
6. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: EUROCRYPT 2016. pp. II: 327–357
7. Bootle, J., Lyubashevsky, V., Nguyen, N.K., Seiler, G.: A non-pcp approach to succinct quantum-safe zero-knowledge. IACR ePrint 2020/737

8. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: IEEE S&P 2018. pp. 315–334
9. Bünz, B., Fisch, B., Szepieniec, A.: Transparent snarks from DARK compilers. In: EUROCRYPT 2020. pp. I: 677–706
10. Cramer, R.: Modular Design of Secure yet Practical Cryptographic Protocols. Ph.D. thesis, CWI and University of Amsterdam (1996)
11. Cramer, R., Damgård, I., Maurer, U.M.: General secure multi-party computation from any linear secret-sharing scheme. In: EUROCRYPT 2000. pp. 316–334
12. Cramer, R., Damgård, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing. Cambridge University Press (2015)
13. Cramer, R., Damgård, I., Pastro, V.: On the amortized complexity of zero knowledge protocols for multiplicative relations. In: ICITS 2012. pp. 62–79
14. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: CRYPTO 1994. pp. 174–187
15. Damgård, I., Fujisaki, E.: A statistically-hiding integer commitment scheme based on groups with hidden order. In: ASIACRYPT 2002. pp. 125–142
16. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: CRYPTO 1986. pp. 186–194
17. Fujisaki, E., Okamoto, T.: Statistical zero knowledge protocols to prove modular polynomial relations. In: CRYPTO 1997. pp. 16–30
18. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. IACR ePrint 2019/953
19. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct nizks without pcps. In: EUROCRYPT 2013. pp. 626–645
20. Groth, J.: Linear algebra with sub-linear zero-knowledge arguments. In: CRYPTO 2008. pp. 192–208
21. Groth, J.: On the size of pairing-based non-interactive arguments. In: EUROCRYPT 2016. pp. II: 305–326. Springer
22. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: ASIACRYPT 2010. pp. 321–340
23. Groth, J., Kohlweiss, M.: One-out-of-many proofs: Or how to leak a secret and spend a coin. In: EUROCRYPT 2015. pp. II: 253–280
24. Hoffmann, M., Klooß, M., Rupp, A.: Efficient zero-knowledge arguments in the discrete log setting, revisited. In: ACM CCS 2019
25. Jivanyan, A., Mamikonyan, T.: Hierarchical one-out-of-many proofs with applications to blockchain privacy and ring signatures. IACR ePrint 2020/430
26. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In: ACM CCS 2019. pp. 2111–2128
27. Naor, M.: On cryptographic assumptions and challenges. In: CRYPTO 2003. pp. 96–109
28. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: CRYPTO 1991. pp. 129–140
29. Wesolowski, B.: Efficient verifiable delay functions. In: EUROCRYPT 2019. pp. III: 379–407
30. Xie, T., Zhang, J., Zhang, Y., Papamanthou, C., Song, D.: Libra: Succinct zero-knowledge proofs with optimal prover computation. In: CRYPTO 2019. pp. III: 733–764