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

Full Version[4,5]- July 16, 2020

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.

1 Introduction

The theory of Σ-Protocols provides a well-understood basis for *plug-and-play* secure algorithmics.[6] Recently, Bulletproofs [BCC+16, BBB+18] have been introduced as a "drop-in replacement" for Σ-Protocols in several important applications. Notably, this includes ZK for arithmetic circuits with communication $O(\log |C| \cdot \kappa)$

[*] thomas.attema@tno.nl

[†] cramer@cwi.nl, cramer@math.leidenuniv.nl

[5] This is the full version of the same title work accepted for publication at CRYPTO 2020.

[5] **Change log** w.r.t. previous version- June 23, 2020: (a) modified the extractor analysis of Appendix A to account for the fact that our previous derivations are only meaningful for a portion of the full parameter space, and (b) for the full range of parameters we rely on prior results and, for this reason, refrain from giving concrete knowledge errors in several theorems.

[6] 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.

bits where $|C|$ is the circuit size[7] 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.[8]

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 [FS86]. 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 (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, for a portion of the full parameter space relevant to our applications, known run-time analyses of knowledge extractors involved and give concrete estimates. For the remainder, we continue to rely on known analyses. 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*

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

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

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* [CDP12], 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 [CDP12]. It is in these adaptations that free choice of linear forms in the pivot is fully exploited; the maps arising from our adaptation of [CDP12] 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 $\mathbf{x}_1, \dots, \mathbf{x}_s$ and a linear form L, allows to open $L(\mathbf{x}_1), \dots, L(\mathbf{x}_s)$ 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 $L_1, \dots, L_s$ 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.*[9]

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 [BCC+16, BBB+18]) but also one derived from the Strong-RSA assumption (as nailed down in a recent work [BFS20] 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 how to base the program on a specific knowledge of exponent assumption.

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

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 [Cra96], it is well known that *special soundness* implies knowledge soundness with knowledge error $1/q$, where q is the size of the challenge set. Depending on a choice for the definition of knowledge soundness, this result can either be shown by an application [Cra96] of Jensen's inequality, or by a more intricate variation of the classical heavy-row type approach [Dam10].

Recently, and particularly for the above mentioned compression techniques, natural generalizations of special soundness have become relevant. However, the mentioned proof techniques are no longer directly applicable. The nature of the compression techniques namely significantly reduces the efficiency of the corresponding knowledge extractors. For this reason prior works [BCC+16, BBB+18] resort to alternative arguments without computing the exact knowledge error. See also [Wik18] and [HKR19] for a discussion on extractor efficiency and knowledge errors.

Here, we show that an adaptation of the proof using Jensen's inequality does apply for a portion of the full parameter space relevant to our applications. This results in a simple proof and an exact knowledge error for this portion of the parameter space. For parameters that do not fall in this range we resort to prior results [BCC+16, BBB+18]. The details of the extractor analysis can be found in Appendix A.

Compressed Pivot with Unconditional Soundness. In addition, we show two approaches for realizing our compressed pivot with *unconditional* 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 Appendix C.

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.

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 [CDP12] 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 [CDM00]. For more information, see Section 12.5.3 in [CDN15] 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 [CDP12] and then we discuss adaptations. The work of [CDP12] 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 $0, m+1, \ldots, 2m$.[10] 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.

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).

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

Our *second observation here* is as follows. Suppose we have an arithmetic circuit[11] C over $\mathbb{Z}_q$ with n inputs, s outputs and m multiplication gates.[12] 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 [CDP12] 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 $n + 2m + 3$.*

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 $O(\kappa \log \gamma)$ 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 and Appendix F.

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

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

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

assumption. More precisely, we show how the polynomial commitment scheme from a recent work [BFS20] can be adapted to open arbitrary linear forms. Our adaptations of the linearization techniques from [CDP12] are directly applicable to the Strong-RSA derived pivot. The details can be found in Section 7 and Appendix G.

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 [Nao03, BCPR14]. 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 [Gro10].

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 [ACF20], we construct logarithmic size proofs of partial knowledge *for all* k, n, by adapting our compressed Σ-protocols and repurposing ideas from [CDS94]. So far, a *direct* [13], linear size solution is known for all k, n [CDS94]; logarithmic size *only* for $k = 1$, i.e., *1-out-of-n proofs* [GK15, BCC$^+$15, JM20]. We note that, for $k = 1$, we nearly halve the best known communication costs.

I. Our program from Lattice Assumptions
From the work of [BLNS20] 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 [BLNS20]. 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 [Gro09], an ingenious recursive approach achieved logarithmic communication complexity [BCC$^+$16]. At its heart lies an earlier version of the BP protocol discussed earlier. Further improvements were introduced in [BBB$^+$18] and later revisited in [HKR19]. Recently, Bünz, Fisch and Szepieniec [BFS20] 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, [BFS20] deploys proofs of exponentiation [Wes19] to reduce the computational complexity.

A common denominator in the aforementioned works is the use of a quadratic constraint as a main pivot. In [Gro09], 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 [BCC$^+$16], except that it also uses an earlier version of the BP idea. In [BBB$^+$18], it is subsequently shown that a modification of the quadratic relation leads to better constants. In [HKR19], more general quadratic constraints were considered with a view towards reducing *computational* complexity in specific ZK scenarios. Also they strive for a more modular approach. However, this induces (minor) communication overhead in comparison to Bulletproofs [BBB$^+$18].

Furthermore, it is worth mentioning that in [BCC$^+$16], 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 [BFS20] bases itself entirely on this polynomial functionality. For

[13] It is possible to formulate a proof of partial knowledge as a circuit ZK problem and solve it with known circuit ZK techniques. However, this results in large arithmetic circuits and for this reason it is interesting to consider direct approaches omitting these circuits.

general relations it uses recent, but complicated, reductions [GWC19, MBKM19, XZZ$^+$19]. 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 [BBB$^+$18] and [HKR19] 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 [HKR19], 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 [CDP12]. The compression is achieved by a suitable adaptation of the BP ideas [BBB$^+$18], and the linearization techniques discard the need for a direct provision to handle nonlinearity. Moreover, plug and play design of applications according to this compressed Σ-protocol theory is just as easy as with the standard Σ-protocol theory. Despite the conceptual simplicity, the communication complexities of our approach are, even including the constants, equal to that of Bulletproofs [BBB$^+$18].

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 Preliminaries

In this section we introduce the basic notation, definitions and conventions used in the remainder of the paper.

Interactive Protocols. Let $R = \{(x, w)\}$ be some NP-relation. Here, x is called a *statement* and w is called a *witness* for x. An interactive protocol Π for relation R is a protocol that allows a prover to convince a verifier that it knows a witness w for given statement x. Protocol 1 shows a graphical protocol description of dummy protocol Π.Protocol Π takes x as public input and w as prover's private input, which we write as either $\Pi(x; w)$ 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, $(x; w) \in R$) is made explicit in the graphical protocol description. An interactive protocol in which the verifier chooses all its messages uniformly at random and independent from the prover's messages is called a *public coin* protocol. All protocols in this work are public coin and can therefore be made non-interactive by applying the Fiat-Shamir transformation [FS86].

Special Soundness and Zero-Knowledge. A public coin protocol is said to be (unconditionally) $(k_1, \dots, k_\mu)$-*special sound* if there exists a polynomial time algorithm that on input a statement x and a $(k_1, k_2, \dots, k_\mu)$-*tree of accepting transcripts*, outputs a witness w for x. See [BCC$^+$16] for a detailed definition. In brief, a $(k_1, k_2, \dots, k_\mu)$-tree of accepting transcripts is a set of $\prod_{i=1}^{\mu} k_i$ accepting transcripts that are arranged in a tree structure. The edges in this three correspond to the verifier's challenges and vertices to the prover's messages, which can be empty. Every node at depth i has precisely k_i children corresponding to k_i pairwise distinct challenges. Every transcript corresponds to exactly one path from the root node to a leaf node. Note that this notion is, in two ways, a natural generalization of the standard notion *special soundness*: (1) from a colliding pair of transcripts to a k-collision and (2) from 1 challenge protocols to protocols with $\mu \geq 1$ challenges. In [BCC$^+$16] it is shown that $(k_1, \dots, k_\mu)$-special soundness implies *witness extended emulation* [Lin03]. A protocol is said to have this property if for any prover $\mathcal{P}^*$ there exists an efficient algorithm, with rewindable oracle access to $\mathcal{P}^*$, that outputs a transcript and, if this transcript is accepting then it outputs, with overwhelming probability, a witness as well. The transcripts generated by this algorithm are required to be indistinguishable from conversations between $\mathcal{P}^*$ and an honest verifier. We show that all

Protocol 1 Dummy Protocol Π for Relation R

Public Parameters : ...

Input$(x; w)$
Output$(y; w')$

$$\begin{array}{ccc} & (x;w) \in R & \\ \textsf{Prover} & & \textsf{Verifier} \\ \dots & \xrightarrow{\hspace{3em}} & \\ & \xleftarrow{\hspace{3em}} & \dots \\ & \vdots & \\ \dots & \xrightarrow{\hspace{3em}} & \dots \end{array}$$

protocols in this work are $(k_1, k_2, \dots, k_\mu)$-special sound for some μ and some set of k_i's. From the result of [BCC+16] it then follows that the protocols in this work are *proofs of knowledge.*

The protocol's public parameters are typically a set of generators $g_1, \dots, g_n, h$ of a group $\mathbb{G}$ of prime order q. We assume that, in the setup phase, these generators are sampled uniformly at random such that the prover does not know a non-trivial DL relation between them. We say that the protocol is *computationally* $(k_1, k_2, \dots, k_\mu)$-special sound, under the DL assumption, if there exists an efficient algorithm that *either* extracts a witness *or* finds a non-trivial DL relation between the public parameters $g_1, \dots, g_n, h$. Protocols that satisfy this computational variant of soundness are also called *arguments of knowledge.* Later we will also consider different set up assumptions, based on Strong-RSA of Knowledge of Exponent derived assumptions. Finally, we will consider standard notions of zero-knowledge such *special honest verifier zero-knowledge* (SHVZK).

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 [Ped91]). *Let $\mathbb{G}$ be an Abelian group of prime order q. Pedersen vector commitments are defined by the following setup and commitment phase:*

– *Setup:* $\mathbf{g} = (g_1, \dots, g_n) \leftarrow_R \mathbb{G}^n$, $h \leftarrow_R \mathbb{G}$.
– *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 $\mathbf{g}^c := (g_1^c, g_2^c, \dots, g_n^c)$ 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 $\mathbf{g} * \mathbf{h} = (g_1h_1, g_2h_2, \dots, g_nh_n)$.

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 $g_1, \dots, g_n, h$.

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

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

where we use the following definition for the set of linear forms on $\mathbb{Z}_q^n$.

Definition 2. $\mathcal{L}\left(\mathbb{Z}_q^n\right) := \{\ (L : Z_q^n \to \mathbb{Z}_q) : L$ *is a* $\mathbb{Z}_q$*-linear map*$\}$.

Protocol 2, 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 [FS86] 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 special sound. Moreover, the communication costs are:*

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

Protocol 2 Σ-protocol Π_0 for relation R
Σ-protocol to prove correctness of a linear form evaluation.

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

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

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

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 Appendix B.

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 [BCC$^+$16] this observation was applied to Groth's Σ-protocol [Gro09]. 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 3 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 special sound, under the discrete logarithm assumption. 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.

Special 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, which completes the proof.

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.

Protocol 3 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$(\widehat{P}, \widehat{L}, \widehat{y}; \widehat{\mathbf{z}})$

$$\widehat{P} = \widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} \in \mathbb{G}$$
$$\widehat{y} = \widehat{L}(\widehat{\mathbf{z}}) \in \mathbb{Z}_q$$

Prover | Verifier

$\xleftarrow{c}$ $\quad c \leftarrow_R \mathbb{Z}_q$

$\xrightarrow{\widehat{\mathbf{z}}}$

$\widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} k^{c\widehat{L}(\widehat{\mathbf{z}})} \overset{?}{=} \widehat{P} k^{c\widehat{y}}$

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 4 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 [BBB+18], protocol Π_2 is unconditionally $(3, \ldots, 3)$-special sound.

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 $(k_1, \ldots, k_\mu)$-special sound, where $k_i = 3$ for all $1 \le i \le \mu$. 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.

Special soundness follows in a similar manner as it does for the amortized Σ-protocol of Theorem 9 (Appendix B). 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.

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

$$V = \begin{pmatrix} 1 & 1 & 1 \\ c_1 & c_2 & c_3 \\ c_1^2 & c_2^2 & c_3^2 \end{pmatrix}, \tag{6}$$

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

$$\begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix}^T := V^{-1} \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^T. \tag{7}$$

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 4 Compressed Proof of Knowledge Π_2 for R_2

Public Parameters : $\widehat{\mathbf{g}}, k$
Input$(Q, \widetilde{L}; \widehat{\mathbf{z}})$
$Q = \widehat{\mathbf{g}}^{\widehat{\mathbf{z}}} k^{\widetilde{L}(\widehat{\mathbf{z}})}$

Prover		Verifier
$A = \widehat{\mathbf{g}}_R^{\widehat{\mathbf{z}}_L} k^{\widetilde{L}_R(\widehat{\mathbf{z}}_L)}$		
$B = \widehat{\mathbf{g}}_L^{\widehat{\mathbf{z}}_R} k^{\widetilde{L}_L(\widehat{\mathbf{z}}_R)}$	$\xrightarrow{A,B}$	
		$c \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c}$	
	$\mathbf{g}' := \widehat{\mathbf{g}}_L^c * \widehat{\mathbf{g}}_R \in \mathbb{G}^{(n+1)/2}$	
	$Q' := AQ^cB^{c^2}$	
	$L' := c\widetilde{L}_L + \widetilde{L}_R$	
$\mathbf{z}' = \widehat{\mathbf{z}}_L + c\widehat{\mathbf{z}}_R$		
if $\left(\mathbf{z}' \in \mathbb{Z}_q^2\right)$:	$\xrightarrow{\mathbf{z}'}$	$(\mathbf{g}')^{\mathbf{z}'} k^{L'(\mathbf{z}')} \overset{?}{=} Q'$
else :	Run $\Pi_2(Q', L'; \mathbf{z}')$ with Public Parameters : $\mathbf{g}', k$	

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 Protocol 5. 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 $(2, 2, k_1, \ldots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. 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.

Special soundness follows from a straightforward combination of the extraction algorithms of protocols Π_0, Π_1 and Π_2.

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 Appendix B.

4.4 Compressed Pivot with Unconditional Soundness

Note that since protocol Π_1 has computational soundness so does the compressed pivot Π_c. In Appendix C we show two approaches for deriving an unconditionally sound compressed pivot.

Protocol 5 Compressed Σ-protocol Π_c for relation R
Compressed Σ-protocol to prove correctness of a linear form evaluation.

$$\textsc{Public Parameters}: \mathbf{g}, h, k$$
$$\textsc{Input}(P, L, y; \mathbf{x}, \gamma)$$

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

Prover		Verifier
$\mathbf{r} \leftarrow_R \mathbb{Z}_q^n, \rho \leftarrow_R \mathbb{Z}_q$ $t = L(\mathbf{r})$ $\widehat{A} = \mathbf{g}^{\mathbf{r}} h^{\rho}$		
	$\xrightarrow{t, \widehat{A}}$	
		$c_0, c_1 \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c_0, c_1}$	
$\mathbf{z} = c_0 \mathbf{x} + \mathbf{r}$ $\phi = c_0 \gamma + \rho$ $\widehat{\mathbf{z}} := (\mathbf{z}, \phi)$		
	$\widehat{\mathbf{g}} := (\mathbf{g}, h)$ $Q := \widehat{A} P^{c_0} k^{c_1(c_0 y + t)}$ $\widetilde{L}(\mathbf{z}, \phi) := c_1 L(\mathbf{z})$	
$A = \widehat{\mathbf{g}}_R^{\widehat{\mathbf{z}}_L} k^{\widetilde{L}_R(\widehat{\mathbf{z}}_L)}$ $B = \widehat{\mathbf{g}}_L^{\widehat{\mathbf{z}}_R} k^{\widetilde{L}_L(\widehat{\mathbf{z}}_R)}$		
	$\xrightarrow{A, B}$	
		$c \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c}$	
	$\mathbf{g}' := \widehat{\mathbf{g}}_L^c * \widehat{\mathbf{g}}_R \in \mathbb{G}^{(n+1)/2}$ $Q' := A Q^c B^{c^2}$ $L' := c\widetilde{L}_L + \widetilde{L}_R$	
$\mathbf{z}' = \widehat{\mathbf{z}}_L + c\widehat{\mathbf{z}}_R$		
if $\left(\mathbf{z}' \in \mathbb{Z}_q^2\right)$:	$\xrightarrow{\mathbf{z}'}$	$(\mathbf{g}')^{\mathbf{z}'} k^{L'(\mathbf{z}')} \overset{?}{=} Q'$
else :	Run $\Pi_2(Q', L'; \mathbf{z}')$ with $\textsc{Public Parameters}: \mathbf{g}', k$	

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 Appendix D.

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. For completeness the diagram of Π_{OPEN} is depicted in Protocol 6. 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 $\Pi_{\text{OPEN}}([\mathbf{x}_1], \dots, [\mathbf{x}_s], L; \mathbf{x}_1, \dots, \mathbf{x}_s)$ 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 soundness properties and communication costs it is implicitly assumed that $[\cdot]$ is instantiated with Pedersen vector commitments and compressed Σ-protocol Π_c.

Protocol 6 Protocol Π_{OPEN} for Opening a Linear Form Evaluated in a Committed Vector
The randomness and generators of the underlying commitment scheme $[\cdot]$ are left implicit.

$\text{INPUT}([\mathbf{x}], L; \mathbf{x})$
$\text{OUTPUT}(L(\mathbf{x}))$

Prover		Verifier
$y = L(\mathbf{x})$	$\xrightarrow{\quad y \quad}$

Run $\Pi_c([\mathbf{x}], L, y; \mathbf{x})$

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 $L_1, \dots, L_s$ and *suppose the prover claims that* $L_i(\mathbf{x}) = 0$ *for* $i = 1 \dots, s$. 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 $\Pi_{\text{NULLITY}}([\mathbf{x}], L_1, \dots, L_s; , \mathbf{x})$ for this protocol. Its diagram is presented in Protocol 7. 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 $\Phi_1, \dots, \Phi_s$, for which we also write $\Pi_{\text{NULLITY}}([\mathbf{x}], \Phi_1, \dots, \Phi_s; \mathbf{x})$. 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 $\Phi_1(\mathbf{x}) - y_1, \dots, \Phi_s(\mathbf{x}) - y_s$. For the interactive protocol that opens an affine map Φ we write $\Pi_{\text{OPEN}}([\mathbf{x}], \Phi; \mathbf{x})$.

Protocol 7 Amortized Protocol Π_{NULLITY} for Many Nullity Checks at Once
The randomness and generators of the underlying commitment scheme $[\cdot]$ are left implicit.

$$\textsc{Input}([\mathbf{x}], L_1, \dots, L_s; \mathbf{x})$$

Prover		Verifier
	$\xleftarrow{\quad\rho\quad}$	$\rho \leftarrow_R \mathbb{Z}_q$

$$L := \sum_{i=1}^{s} L_i \rho^{i-1}$$

$$\text{Run } \Pi_c([\mathbf{x}], L, 0; \mathbf{x})$$

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 Appendix E.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 Appendix E.2. 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 $1, \dots, n$ of the inputs and an ordering $1, \dots, m$ of the multiplication gates.

The approach is to combine the compressed pivot with an adaptation of the work of [CDP12] 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 [CDM00].[14] 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. More precisely, we construct a ZK protocol for the following circuit satisfiability relation:

$$R_{cs} = \{(C; \mathbf{x}) : C(\mathbf{x}) = 0\}. \tag{8}$$

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

[14]For a general description of efficient ZK verification of secret multiplications, in terms of (strongly-multiplicative) arithmetic secret sharing, see Section 12.5.3 [CDN15].

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 [CDP12]. 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*[15] $u_i, v_i : \mathbb{Z}_q^{n+m} \to \mathbb{Z}_q$, 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,

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

$\rho := f(0)$, 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 8 and denoted by Π_{cs}. Protocol Π_{cs} only requires black-box access to the commitment scheme $[\cdot]$. For notational convenience, we write

$$\Pi_{\text{NULLITY}}\left([\mathbf{y}], C(\mathbf{x}), f(c) - y_1, g(c) - y_2, h(c) - y_3; \mathbf{y}\right) \tag{9}$$

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 special soundness property 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 witness extended emulation follows from the special soundness property of Π_{cs}.

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 $(2m + 1, s + 3, 2, 2, k_1, \ldots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. 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.

Special 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 $(s + 3)$-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 special soundness claim now from the properties of protocol Π_c.

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.

Protocol 8 Circuit Satisfiability Argument $\mathit{\Pi}_{cs}$ for Relation R_{cs}
The polynomials f and g are sampled uniformly at random such that their evaluations in $1, \dots, m$ coincide with the left and, respectively, right inputs of the m multiplication gates of C evaluated at $\mathbf{x}$.

$$\textsc{Input}(C; \mathbf{x})$$

$$C : \mathbb{Z}_q^n \to \mathbb{Z}_q^s \qquad C(\mathbf{x}) = 0$$

$$\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
f, g \leftarrow_R \mathbb{Z}_q[X]_{\leq m} & & \\
h(X) := f(X)g(X) & & \\
\mathbf{y} = (\mathbf{x}, f(0), g(0), h(0), & \xrightarrow{[\mathbf{y}]} & \\
h(1), \dots, h(2m)) & & \\
 & & c \leftarrow_R \mathbb{Z}_q \setminus \{1, \dots, m\} \\
z_1 = f(c) & \xleftarrow{c} & \\
z_2 = g(c) & & \\
z_3 = h(c) & \xrightarrow{z_1, z_2, z_3} & \\
 & & z_3 \overset{?}{=} z_1 z_2
\end{array}$$

$$\mathit{\Pi}_{\textsc{Nullity}} \left([\mathbf{y}], \begin{array}{c} C(\mathbf{x}) \\ f(c) - z_1 \\ g(c) - z_2 \\ h(c) - z_3 \end{array} ; \mathbf{z} \right)$$

Theorem 5 (Circuit ZK Case 1). *$\mathit{\Pi}_{cs}^{(1)}$ 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 $(2m+1, \max(n, s+3)+1, 2, 2, 3, 2, k_1, \dots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. Moreover, the communication costs are:*

- *$\mathcal{P} \to \mathcal{V}$: $2 \lceil \log_2(n + 2m + 6) \rceil + 4$ 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 $\mathit{\Pi}_{cs}^{(2)}$ 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). *$\mathit{\Pi}_{cs}^{(2)}$ 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 $(2m + 1, n + 1, s + 4, 2, 2, k_1, \dots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. 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 [BCC+16, BBB+18]. Note that this black-box approach for range proofs can also be instantiated from the circuit ZK protocols of (e.g.) [BCC+16] and [BBB+18]. For details we refer to Appendix F.

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 Appendix G and [BFS20] 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 [FO97, DF02]. 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 [FO97, DF02] 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 [BFS20]. 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 [BFS20] 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, [BFS20] deploys Proofs of Exponentiation (PoE) [Wes19] 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 [BFS20] 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 [BP97]. 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 DL or strong-RSA assumption. KEA is not an intractability assumption and it is unfalsifiable [Nao03, BCPR14]. 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 [Gro10] 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 $h' := h^\alpha, g_1' := g_1^\alpha, \dots, g_n' := g_n^\alpha$. 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 [Gro10]. 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 $\mathbf{a} = (a_1, \dots, a_n) \in \mathbb{Z}_q^n$ 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:

$$f(Y) := \gamma + \sum_{i=1}^{n} x_i Y^i, \quad g(Y) := \sum_{i=0}^{n-1} a_{n-i} Y^i, \quad h(Y) := f(Y)g(Y) = \sum_{i=0}^{2n-1} c_i Y^i.$$

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 [Gro10, 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 [Gro10].

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 [GGPR13, Gro16]. 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 thank Serge Fehr, Toon Segers and Thijs Veugen for extensive commenting at an early stage. We also thank Jens Groth for useful editorial comments and pointers. We are grateful for a comment by Michael Klooß that our exact analysis of knowledge error is only meaningful for a portion of the full parameter space relevant to our application. 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 project QSC.

References

ACF20. Thomas Attema, Ronald Cramer, and Serge Fehr. Compressing proofs of k-out-of-n-partial knowledge. *IACR Cryptol. ePrint Arch.*, 2020:753, 2020.

BBB$^+$18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society, 2018.

BCC$^+$15. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short accountable ring signatures based on DDH. In *ESORICS (1)*, volume 9326 of *Lecture Notes in Computer Science*, pages 243–265. Springer, 2015.

BCC$^+$16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT (2)*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, 2016.

BCPR14. Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. On the existence of extractable one-way functions. In *STOC*, pages 505–514. ACM, 2014.

BFS20. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from DARK compilers. In *EUROCRYPT (1)*, volume 12105 of *Lecture Notes in Computer Science*, pages 677–706. Springer, 2020.

BG92. Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *CRYPTO*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer, 1992.

BLNS20. Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. A non-pcp approach to succinct quantum-safe zero-knowledge. *IACR Cryptol. ePrint Arch.*, page 737, 2020.

BN06. Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *ACM Conference on Computer and Communications Security*, pages 390–399. ACM, 2006.

BP97. Niko Baric and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494. Springer, 1997.

CDM00. Ronald Cramer, Ivan Damgård, and Ueli M. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 316–334. Springer, 2000.

CDN15. Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

CDP12. Ronald Cramer, Ivan Damgård, and Valerio Pastro. On the amortized complexity of zero knowledge protocols for multiplicative relations. In *ICITS*, volume 7412 of *Lecture Notes in Computer Science*, pages 62–79. Springer, 2012.

CDS94. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 1994.

Cra96. Ronald Cramer. *Modular Design of Secure yet Practical Cryptographic Protocols*. PhD thesis, CWI and University of Amsterdam, 1996.

Dam10. Ivan Damgård. On sigma-protocols. Lecture Notes, Aarhus University, Department of Computer Science, 2010.

DF02. Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142. Springer, 2002.

FO97. Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 1997.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

GGPR13. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct nizks without pcps. In *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645. Springer, 2013.

GK15. Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In *EUROCRYPT (2)*, volume 9057 of *Lecture Notes in Computer Science*, pages 253–280. Springer, 2015.

Gro09. Jens Groth. Linear algebra with sub-linear zero-knowledge arguments. In *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 192–208. Springer, 2009.

Gro10. Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 321–340. Springer, 2010.

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In *EUROCRYPT (2)*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2016.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptol. ePrint Arch.*, 2019:953, 2019.

HKR19. Max Hoffmann, Michael Klooß, and Andy Rupp. Efficient zero-knowledge arguments in the discrete log setting, revisited. In *ACM Conference on Computer and Communications Security*, pages 2093–2110. ACM, 2019.

HM98. Shai Halevi and Silvio Micali. More on proofs of knowledge. *IACR Cryptol. ePrint Arch.*, 1998:15, 1998.

JM20. Aram Jivanyan and Tigran Mamikonyan. Hierarchical one-out-of-many proofs with applications to blockchain privacy and ring signatures. *IACR Cryptol. ePrint Arch.*, 2020:430, 2020.

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

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In *ACM Conference on Computer and Communications Security*, pages 2111–2128. ACM, 2019.

Nao03. Moni Naor. On cryptographic assumptions and challenges. In *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 96–109. Springer, 2003.

Ped91. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

Wes19. Benjamin Wesolowski. Efficient verifiable delay functions. In *EUROCRYPT (3)*, volume 11478 of *Lecture Notes in Computer Science*, pages 379–407. Springer, 2019.

Wik18. Douglas Wikström. Special soundness revisited. *IACR Cryptol. ePrint Arch.*, 2018:1157, 2018.

XZZ$^+$19. Tiancheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. In *CRYPTO (3)*, volume 11694 of *Lecture Notes in Computer Science*, pages 733–764. Springer, 2019.

A Extractor Analysis

This appendix describes an extractor analysis for a generalized notion of special soundness: $(k_1, \ldots, k_\mu)$-*special soundness*. Compared to standard special soundness this is a generalization in two ways: (1) from requiring a colliding pair of transcripts to requiring a k-collision of k transcripts and (2) from protocols containing 1 challenge, sent from the verifier to the prover, to protocols containing μ challenges. We show that $(k_1, \ldots, k_\mu)$-special soundness implies, in a portion of the full parameter space, *knowledge soundness*. Moreover, we recall that the result that shows that $(k_1, \ldots, k_\mu)$-special soundness implies witness extended emulation.

An interactive protocol Π for relation R is said to be *knowledge sound* with knowledge error $\kappa(\cdot)$ if the following holds [BG92]. There exists an algorithm χ that, on input a statement x and given rewindable oracle access to a (possibly dishonest) prover $\mathcal{P}^*$, outputs a witness w for x in expected time $O\left(1/(\epsilon(x) - \kappa(x))\right)$, where $\epsilon(x) > \kappa(x)$ is the probability that the verifier accepts a protocol execution with $\mathcal{P}^*$ on public input x. Here, invoking $\mathcal{P}^*$ is assumed to take unit time. Such an algorithm is called a *knowledge extractor*.

An alternative definition of knowledge soundness [HM98] requires the knowledge extractor to run in *strict* polynomial time and to succeed with probability at least $(\epsilon(x) - \kappa(x))^c$ for some constant c. Note that by repeating such an extractor until it succeeds a new extractor with expected run time $O(1/(\epsilon(x) - \kappa(x))^c)$ is obtained. This alternative definition, and in particular the constant c, therefore allows an extractor to be somewhat less efficient. An interactive protocol that is knowledge sound is said to be a *proof of knowledge* (PoK).

However, a technical issue with both these definitions for knowledge soundness arises when proofs of knowledge are used within larger cryptographic protocols. To avoid these composition issues, Lindell [Lin03] introduced the notion *witness extended emulation*. An interactive protocol is said to have witness extended emulation if there exists an expected polynomial time algorithm that, on input a statement x and given rewindable oracle access to a (possibly dishonest) prover $\mathcal{P}^*$, not only outputs a witness w but also a transcript that is statistically indistinguishable from a conversation between $\mathcal{P}^*$ and an honest verifier. Moreover, the probability that the transcript is accepting and that the witness is not valid is negligible. Such an algorithm is called a *witness extended emulator* and interactive protocols that have witness extended emulation are also called proofs of knowledge.

When $\mathcal{P}^*$ succeeds only with negligible probability knowledge extractors are, in contrast to witness extended emulators, not required to run in expected polynomial time. However, a witness extended emulator is allowed to fail in outputting a valid witness with negligible probability. For this reason, a witness extended emulator is only required to output witnesses for provers $\mathcal{P}^*$ that succeed with non-negligible probability.

In Section A.1, we first show that any k-special sound protocol has a strict polynomial time knowledge extractor that succeeds with probability at least $(\epsilon(x) - (k-1)/q)^k$, where q is the size of the challenge set. The argument follows by an application of Jensen's inequality. Second, by a more intricate heavy-row type analysis, we also show the existence of an extractor that output a witness in expected time $O\left(1/\left(\epsilon(x)-(k-1)/q\right)\right)$.

In Section A.2, we show that any $(k_1,\dots,k_\mu)$-special sound protocol has a strict polynomial time knowledge extractor that succeeds with probability at least $(\epsilon(x)-\kappa(x))^K$, where $K=\prod_{i=1}^{\mu}k_i$ and

$$\kappa=\frac{\sum_{i=1}^{\mu}(k_i-1)q^{\mu-i}\prod_{j=1}^{i-1}(q-k_j+1)}{q^{\mu}}. \tag{10}$$

Since the success probability of this extractor degrades exponentially with K, this result only implies a meaningful definition of knowledge soundness when K is constant in the security parameter. Therefore this result is only applicable to a portion of the full parameter space relevant to our applications. It turns out to be a challenging task to construct efficient knowledge extractors and find exact knowledge errors for protocols that satisfy this generalized notion of special soundness. See also [Wik18] and [HKR19] for a discussion on knowledge errors and extractor efficiencies. For this reason, we resort to prior work [BCC$^+$16] that shows that any $(k_1,\dots,k_\mu)$-special sound protocol has witness extended emulation, and omit determining exact knowledge errors. For completeness, we recall this result in Lemma 5.

A.1 *k*-Special Soundness

In the theory of Σ-protocols, an interactive protocol for relation R is called *special sound* if there exists an efficient algorithm that extracts a witness w for statement x from a "collision," i.e., two accepting conversations (x,a,c,z) and (x,a,c',z') with $c\neq c'$. It is well known that special soundness implies both definitions of knowledge soundness [BG92, HM98] with knowledge error $1/q$, where q is the size of the challenge set.

Let us first consider the second definition of knowledge soundness and show the existence of a polynomial time extractor that succeeds with probability at least $(\epsilon(x)-1/q)^2$. This result [Cra96] can be shown to follow from an application of Jensen's inequality to the convex function $f(X)=X(X-1/q)$. We note that [BN06] considers extraction for a more general class of randomized algorithms, i.e., a class that is not restricted to Σ-protocols. Their proof only requires a minor adaptation of the techniques from [Cra96]. However, this generalization is not sufficient for our purposes.

To show that a special sound protocol is knowledge sound, the following "collision-game" is defined in [Cra96]. This is essentially the game played by the knowledge extractor and Lemma 2 gives a bound on the success probability when playing this game. Both the game and the lemma are almost identical to the ones found in [Cra96].

Consider a 0/1-matrix with n rows and q columns. The rows correspond to the prover's randomness and the columns to the verifier's randomness. An entry of the matrix is 1 if the prover is able to supply an accepting response for the associated first message and challenge and 0 otherwise. Let ϵ denote the number of ones in H.

The game goes as follows. Select an entry of H uniformly at random. If this entry is a 1, select another entry of the same row uniformly at random. If this entry is again a 1 the game outputs success.

To bound the success probability of the collision-game, Jensen's inequality is used. Jensen's inequality states that if X is a real random variable and f is a continuous convex function defined on the support of X, it holds that

$$f\left(\mathbb{E}[X]\right)\leq\mathbb{E}[f(X)]. \tag{11}$$

Lemma 2 (Lemma 2.1 of [Cra96]). *Let H be a 0/1-matrix with n rows and q columns, and let ϵ denote the fraction of 1-entries in H. Suppose $\epsilon>1/q$. Then the success probability of one iteration of the "collision-game" is greater than or equal to $\epsilon(\epsilon-1/q)$.*

Proof. Let e_i denote the number of 1-entries in the i-th row, $i = 1 \dots n$. and let ϵ_i denote the fraction of 1-entries in the i-th row, i.e., $\epsilon_i = e_i/q$. Clearly, the success-probability is equal to[16]

$$\frac{1}{n}\sum_{i=1}^{n} \epsilon_i \left(\frac{q\epsilon_i - 1}{q-1}\right) = \frac{1}{n}\frac{q}{q-1}\sum_{i=1}^{n} \epsilon_i \left(\epsilon_i - \frac{1}{q}\right). \tag{12}$$

Now observe that $\mathbb{E}[\epsilon_i] = \epsilon$, put $f(x) = x(x - 1/q)$ on the interval $[0,1]$ and apply Jensen's inequality. Together with the fact that $q/(q-1) > 1$ the Lemma follows.

Using this lemma, it is straightforward to construct a polynomial time knowledge extractor that succeeds with probability at least $(\epsilon(x) - 1/q)^2$. Instead, we immediately consider a generalization that has recently become relevant, k-special soundness. A 3-move interactive protocol is called k-special sound, if there exists an efficient algorithm that takes as input k accepting conversations $(x, a, c_1, z_1), \dots, (x, a, c_k, z_k)$ with $c_i \neq c_j$, $\forall i \neq j$, and outputs a witness w for x. This set of k accepting transcripts with common first message and pairwise distinct challenges is called a *k-collision.* The proof technique using Jensen's inequality is no longer directly applicable, since the associated function is not convex. Here, we show that an adaptation of the proof using Jensen's inequality does apply. To this end let us consider the following function.

$$f : \mathbb{R} \to \mathbb{R} : \quad x \mapsto \begin{cases} \prod_{j=0}^{k-1} \frac{q}{q-j}\left(x - \frac{j}{q}\right), & \text{if } x \geq \frac{k-1}{q}, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

Recall that $q = |\mathcal{C}|$.

It is easily seen that f is twice-differentiable and $f''(x) \geq 0$ for all $x \in \mathbb{R} \setminus \left\{\frac{k-1}{q}\right\}$. Moreover, for $x_0 = \frac{k-1}{q}$ it holds that

$$\lim_{x \uparrow x_0} \frac{f(x) - f(x_0)}{x - x_0} = 0 \leq \frac{q}{q-k+1}\prod_{j=0}^{k-2}\frac{k-1-j}{q-j} = \lim_{x \downarrow x_0}\frac{f(x) - f(x_0)}{x - x_0}. \tag{14}$$

Hence, f is a convex function.

The following lemma now shows the existence of a polynomial time extractor that succeeds with probability at least $(\epsilon(x) - (k-1)/q)^k$. If k is constant in the security parameter, it therefore follows that k-special soundness implies knowledge soundness according to the definition of [HM98].

Theorem 7 (Knowledge Soundness Type II). *Let $(\mathcal{P}, \mathcal{V})$ be a k-special sound interactive protocol for relation R and let x be some statement, where $\mathcal{V}$ samples the challenge uniformly at random from a challenge set of size q. Let $\mathcal{P}^*$ be a prover such that $(\mathcal{P}^*, \mathcal{V})$ accepts with probability $\epsilon(x) > \frac{k-1}{q}$. Then there exists a polynomial time extractor $\mathcal{E}$ with rewindable black-box access to $\mathcal{P}^*$ that on input x outputs a witness w for x with probability at least*

$$\prod_{j=0}^{k-1}\left(\epsilon(x) - \frac{j}{q}\right) \geq \left(\epsilon(x) - \frac{k-1}{q}\right)^k, \tag{15}$$

in at most k calls to $\mathcal{P}^*$.

Proof. $\mathcal{E}$ runs $(\mathcal{P}^*, \mathcal{V})$ on a random challenge $c \in \mathcal{C}$. If $\mathcal{V}$ accepts, $\mathcal{E}$ rewinds to move 2 and samples a uniform random challenge from $\mathcal{C} \setminus \{c\}$. $\mathcal{E}$ continues until it aborts or has extracted k accepting transcripts. In the latter case, k-special soundness implies the existence of an efficient algorithm to compute a witness w. So let us now determine the success probability of $\mathcal{E}$.

Let a be any first message of $(\mathcal{P}^*, \mathcal{V})$ on input $(x; w) \in R$. Let ϵ_a be the probability that $\mathcal{P}^*$ succeeds conditioned on the first message being equal to a. Then $\mathbb{E}[\epsilon_a] = \epsilon(x)$, where the expected value is taken over all possible first messages a.

[16]This is minor correction of the original proof, which incorrectly states that the success probability is equal to $\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\left(\epsilon_i - \frac{1}{q}\right)$.

Moreover, the success probability of $\mathcal{E}$, conditioned on the first message being equal to a can easily seen to be equal to $f(\epsilon_a)$, where f is defined in Equation 13.

Hence, the unconditional success probability of $\mathcal{E}$ equals

$$\mathbb{E}[f(X)] \geq f\left(\mathbb{E}[X]\right) = f(\epsilon(x)) \geq \prod_{j=0}^{k-1}\left(\epsilon(x) - \frac{j}{q}\right), \tag{16}$$

where the first inequality follows from Jensen's inequality.

The success probability of this extractor of Theorem 7 degrades exponentially with k. For this reason, this extractor only implies a meaningful definition of knowledge soundness if k is constant in the security parameter. For this reason we also consider a more intricate heavy-row type extraction algorithm. The extractor is an adaptation of the knowledge extractor of [Dam10] that merely considered 2-special-soundness. Its properties are summarized in the following theorem. In particular, it follows that k-special soundness implies knowledge soundness according to the definition of [BG92].

Theorem 8 (Knowledge Soundness Type I). *Let $(\mathcal{P},\mathcal{V})$ be a k-special sound interactive protocol for relation R and let x be some statement, where $\mathcal{V}$ samples the challenge uniformly at random from a challenge set of size q. Let $\mathcal{P}^*$ be a prover such that $(\mathcal{P}^*,\mathcal{V})$ accepts with probability $\epsilon(x) > \frac{k-1}{q}$. Then there exists a polynomial time extractor $\mathcal{E}$ with rewindable black-box access to $\mathcal{P}^*$ that on input x outputs a witness w for x in expected time:*

$$O\left(\frac{k}{\epsilon(x) - (k-1)/q}\right). \tag{17}$$

Here, invoking $\mathcal{P}^*$ is assumed to take unit time.

Proof. We aim to construct an algorithm that outputs a k-collision for protocol $(\mathcal{P},\mathcal{V})$ and public input x. Then k-special soundness implies the existence of an efficient extractor. The algorithm is required to run in expected time $O\left(\frac{k}{\epsilon(x)-(k-1)/q}\right)$. We now describe the algorithm and analyze its expected run time.

The algorithm considers two different cases that depend on the success probability $\epsilon(x)$ and for each case we describe a different approach. Because the success probability $\epsilon(x)$ is unknown, the extractor runs both approaches in parallel.

Case 1. Let us first consider the case $(k-1)/q < \epsilon(x) < 4(k-1)/q$, i.e., $\epsilon(x) = (1+\delta)\frac{k-1}{q}$ for some $0 < \delta < 3$. The algorithm starts by generating random transcripts by invoking $\mathcal{P}^*$ until it finds an accepting transcripts (a, c, z). This takes expected time $1/\epsilon(x)$. Let ϵ_a be the success probability of $\mathcal{P}^*$ conditioned on the prover sending first message a. Note that, if $\epsilon_a \geq k/q$ there exist at least k accepting transcripts with common first message a and pairwise distinct challenges.

To find these transcripts, the algorithm rewinds $q-1$ times and tries all challenges $c' \neq c$ given first message a. This second step takes time $q-1$. As a result the algorithm has generated q transcripts, for which at least the first one is accepting, in expected time

$$\frac{1}{\epsilon(x)} + q - 1 < \frac{1}{\epsilon(x)} + \frac{k-1}{(k-1)/q} = \frac{1+(1+\delta)(k-1)}{\epsilon(x)}. \tag{18}$$

Moreover, the algorithm is successful if and only if at least k of these transcripts are accepting, i.e., if and only if $\epsilon_a \geq k/q$. Let us now determine this success probability. Let R be the number of possible random choices by $\mathcal{P}^*$ and recall that the number of possible random choices by $\mathcal{V}$ is q. For precisely $\epsilon(x)Rq$ of these random choices $\mathcal{P}^*$ succeeds. Moreover, at least $\epsilon(x)Rq - (k-1)R = (k-1)R\delta$ of these successful choices correspond to a first message a with $\epsilon_a \geq k/q$. Hence, the success probability of this algorithm is at least,

$$\frac{(k-1)R\delta}{\epsilon(x)Rq} = \frac{\delta}{1+\delta}. \tag{19}$$

Repeating this algorithm until it finds a sets of k accepting transcripts therefore results in an expected run time of,

$$\frac{(1+\delta)\,(1+(1+\delta)(k-1))}{\delta\epsilon(x)} = \frac{1+(1+\delta)(k-1)}{\delta(k-1)/q} = \frac{1+(1+\delta)(k-1)}{\epsilon(x)-(k-1)/q} < \frac{4k-3}{\epsilon(x)-(k-1)/q}, \tag{20}$$

which is $O\left(\frac{k}{\epsilon(x)-(k-1)/q}\right)$ and therefore proves the theorem for this case.

Case 2. Let us now consider the case $\epsilon(x) \geq 4(k-1)/q$, i.e., $\epsilon(x) = (1+\delta)\frac{k-1}{q}$ or some $\delta \geq 3$. In this case, the previous approach of simply rewinding $q-1$ times does not work since we cannot bound δ in Equation 20. For this reason, we must abort the rewinding procedure at some point to reduce the expected run time.

The algorithm starts as before by generating random transcripts until it finds an accepting transcript (a,c,z). By a simple counting argument it can be seen that $\epsilon_a \geq \epsilon(x)/2$ with probability at least $1/2$ [Dam10]. Subsequently, the algorithm rewinds and tries different challenges until it is told to abort.

The abort procedure works as follows. After every rewinding step the algorithm flips a coin that lands heads with probability $\epsilon(x)/(8(k-1))$. If the coin lands tails the algorithm continues by rewinding again and if not the algorithm aborts. The extractor can implement such a coin, even if it does not know $\epsilon(x)$, by querying $\mathcal{P}^*$. The expected run time of this step is $8(k-1)/\epsilon(x)$. The probability that the algorithm rewinds at most $4(k-1)/\epsilon(x)$ times can, by a union bound, by upper bounded by $1/2$. In other words, the probability that the algorithm rewinds at least $4(k-1)/\epsilon(x)$ times is at least $1/2$.

Let us now determine the probability that the algorithm finds at least k accepting transcripts conditioned on $\epsilon_a \geq \epsilon(x)/2$ and the event that the algorithm rewinds at least $4(k-1)/\epsilon(x)$ times. This conditional success probability is at least the probability that the algorithm succeeds after exactly $4(k-1)/\epsilon(x)$ rewinds.

The rewinding procedure can be modeled by a negative hypergeometric distribution with population size $q-1$ from which draw without replacement until we have found $k-1$ accepting transcripts out of $\epsilon_a q - 1 \geq \epsilon(x)q/2 - 1 \geq 2(k-1)-1 \geq k-1$ transcripts. The expected number of rewinds before obtaining k accepting transcripts is therefore equal to,

$$\frac{k-1}{\epsilon_a} \leq \frac{2(k-1)}{\epsilon(x)}. \tag{21}$$

By Markov's inequality the probability that we need more than $4(k-1)/\epsilon(x)$ rewinds is at most $1/2$. Therefore, conditioned on $\epsilon_a \geq \epsilon(x)/2$ and the event that the algorithm rewinds at least $4(k-1)/\epsilon(x)$ times, the algorithm succeeds in obtaining k accepting transcripts with probability at least $1/2$.

Altogether the case 2 algorithm runs in expected time at most,

$$\frac{1}{\epsilon(x)} + \frac{8(k-1)}{\epsilon(x)}, \tag{22}$$

which is $O\left(\frac{k}{\epsilon(x)-(k-1)/q}\right)$. Moreover, it succeeds in finding k accepting transcripts, with common first message a and pairwise distinct challenges c_i, with probability at least $1/8$. Hence, by running this algorithm an expected number of 8 times gives the desired extraction algorithm, which concludes the proof of this theorem.

A.2 $(k_1,\dots,k_\mu)$-Special Soundness

Recall that an interactive protocol is called $(k_1,\dots,k_\mu)$-special sound, if there exists an efficient algorithm that computes a witness from any set of $K := \prod_{i=1}^{\mu} k_i$ accepting transcripts $(x, a, c_{1,j}, z_{1,j}, \dots, c_{\mu,j}, z_{\mu,j})$, $1 \leq j \leq K$, that they are in a $(k_1,\dots,k_\mu)$-tree structure. The vertices of this tree correspond to the prover's messages, in particular the root of a $(k_1,\dots,k_\mu)$-tree is some message a. The edges of the tree correspond to the verifier's challenges. Every vertex at depth i has precisely k_i children corresponding to k_i pairwise distinct challenges. This way we obtain precisely K paths from the leaves to the root representing the accepting transcripts. We also note that in some protocols it occurs that the verifier sends multiple challenges in one message. In these protocol the number of moves is strictly less than $2\mu + 1$. However, we still consider depth μ trees by allowing prover's messages to be empty.

In this section, we show the existence of a polynomial time extractor that succeeds with probability $(\epsilon(x) - \kappa)^K$, where $K = \prod_{i=1}^{\mu} k_i$ and κ is defined in Equation 23. For notational convenience, Lemma 3 assumes that all challenges are sampled from $\mathbb{Z}_q$ uniformly at random. Subsequently, Lemma 4 generalizes this to the case where the verifier samples from different subsets of $\mathbb{Z}_q$ in the different rounds of the protocol. The bound $\epsilon(x) > \kappa$ is tight, since the existence of an extractor, not necessarily expected polynomial time, can not be guaranteed if $\epsilon(x) \leq \kappa$. However, the success probability of this extractor degrades exponentially with K. Therefore this extractor only implies knowledge soundness if K is constant in the security parameter. For this reason, we also recall, in Lemma 5, a result of [BCC+16]. This result shows that a $(k_1, \dots, k_\mu)$-special sound protocol has witness extended emulation and it follows from the extractor analysis of [BCC+16, Lemma 1].

Lemma 3. *Let $(\mathcal{P}, \mathcal{V})$ be a $(k_1, \dots, k_\mu)$-special sound $(2\mu+1)$-move interactive protocol for relation R, where $\mathcal{V}$ samples each challenge uniformly at random from a challenge set of size q. Let x be some statement. Let $\mathcal{P}^*$ be a prover such that $(\mathcal{P}^*, \mathcal{V})$ accepts with probability $\epsilon(x) > \kappa$, where*

$$\kappa = \frac{\sum_{i=1}^{\mu}(k_i - 1)q^{\mu-i}\prod_{j=1}^{i-1}(q - k_j + 1)}{q^\mu} \leq \frac{\sum_{i=1}^{\mu}(k_i - 1)}{q}. \tag{23}$$

Then there exists a polynomial time extractor $\mathcal{E}$ with rewindable black-box access to $\mathcal{P}^*$ that on input x outputs a witness w for x with probability at least $(\epsilon(x) - \kappa)^K$ in at most K calls to $\mathcal{P}^*$, where $K = \prod_{i=1}^{\mu} k_i$.

Proof. We construct a polynomial time algorithm that generates a $(k_1, \dots, k_\mu)$-tree of accepting transcripts with probability at least $(\epsilon(x) - \kappa)^K$ in at most K calls. The lemma then follows from the definition of $(k_1, \dots, k_\mu)$-special soundness.

For $0 \leq m \leq \mu - 1$ and $c_i \in \mathcal{C}$ let $\textsc{Tree}(x, a, c_1, \dots, c_m)$ be the algorithm that tries to find a $(k_{m+1}, \dots, k_\mu)$-sub-tree after the first $2m + 1$ rounds have been fixed by $a, c_1, \dots, c_m$. More precisely, for $m = \mu$ it simply runs $\mathcal{P}^*$ on challenges $c_1, \dots, c_\mu$ and for $m < \mu$ it runs $\textsc{Tree}(x, a, c_1, \dots, c_m, y_\ell)$ for $1 \leq \ell \leq k_{m+1}$ and $y_\ell \in \mathcal{C}$ sampled uniformly at random such that $y_i \neq y_j$ for all $i \neq j$. We say $\textsc{Tree}$ aborts if at any stage the verifier $\mathcal{V}$ rejects and write $\textsc{Tree} = \perp$ in this case.

For notational convenience we define $\bar{\mathbf{c}}_m := (x, a, c_1, \dots, c_m)$. For such a vector we define $\epsilon_{\bar{\mathbf{c}}_m}$ to be the probability that $\mathcal{P}^*$ succeeds conditioned on the first $2m + 1$ rounds to coincide with $\bar{\mathbf{c}}_m$. Moreover, let us define

$$\kappa_m := \frac{\sum_{i=m+1}^{\mu}(k_i - 1)q^{\mu-i}\prod_{j=m+1}^{i-1}(q - k_j + 1)}{q^{\mu-m}}. \tag{24}$$

Finally, we let $K_m = \prod_{i=m+1}^{\mu} k_i$. We will show by induction that the success probability P_m of $\textsc{Tree}(x, a, c_1, \dots, c_m)$ is at least $\max(\epsilon_{\bar{\mathbf{c}}_m} - \kappa_m, 0)^{K_m}$ for all $0 \leq m \leq \mu$.

For $m = \mu$ the induction hypothesis immediately follows by the definition of $\epsilon_{\bar{\mathbf{c}}_m}$. So let us assume that the success probability of $\textsc{Tree}(x, a, c_1, \dots, c_m)$ is at least $\max(\epsilon_{\bar{\mathbf{c}}_m} - \kappa_m, 0)^{K_m}$ for all $m > M$. Then,

$$\begin{aligned}
P_M &:= P\left(\textsc{Tree}(x, a, c_1, \dots, c_M) \neq \bot\right), \\
&= \prod_{\ell=1}^{k_{M+1}} P\left(\textsc{Tree}(x, a, c_1, \dots, c_M, y_\ell) \neq \bot\right), \\
&\geq \prod_{\ell=1}^{k_{M+1}} \max\left(\epsilon_{\bar{\mathbf{c}}_M, y_\ell} - \kappa_{M+1}, 0\right)^{K_{M+1}}, \\
&\overset{(1)}{\geq} \prod_{\ell=1}^{k_{M+1}} \max\left(\frac{q}{q-\ell+1}\left(\epsilon_{\bar{\mathbf{c}}_M} - \frac{\ell-1}{q}\right) - \kappa_{M+1}, 0\right)^{K_{M+1}}, \\
&= \prod_{\ell=1}^{k_{M+1}} \max\left(\frac{q}{q-\ell+1}\left(\epsilon_{\bar{\mathbf{c}}_M} - \frac{\ell-1+\kappa_{M+1}(q-\ell+1)}{q}\right), 0\right)^{K_{M+1}}, \\
&\overset{(2)}{\geq} \prod_{\ell=1}^{k_{M+1}} \left(\epsilon_{\bar{\mathbf{c}}_M} - \frac{k_{M+1}-1+\kappa_{M+1}(q-k_{M+1}+1)}{q}, 0\right)^{K_{M+1}}, \\
&= \max\left(\epsilon_{\bar{\mathbf{c}}_M} - \frac{k_{M+1}-1+\kappa_{M+1}(q-k_{M+1}+1)}{q}, 0\right)^{K_M}, \\
&\overset{(3)}{=} \max\left(\epsilon_{\bar{\mathbf{c}}_M} - \kappa_M, 0\right)^{K_M}.
\end{aligned} \tag{25}$$

For inequality (1) we use that y_ℓ is sampled uniformly at random from $\mathcal{C} \setminus \{y_1, \dots, y_{\ell-1}\}$, hence

$$\begin{aligned}
\epsilon_{\bar{\mathbf{c}}_M, y_\ell} &= \frac{q}{q-\ell+1}\left(\epsilon_{\bar{\mathbf{c}}_M} - \frac{1}{q}\sum_{j=1}^{\ell-1} \epsilon_{\bar{\mathbf{c}}_M, y_j}\right), \\
&\geq \frac{q}{q-\ell+1}\left(\epsilon_{\bar{\mathbf{c}}_M} - \frac{\ell-1}{q}\right).
\end{aligned} \tag{26}$$

For inequality (2) we use that $\ell \leq k_{M+1}$ and that $0 \leq \kappa_m \leq 1$ for all m. For equality (3) we use that

$$\frac{k_{M+1}-1+\kappa_{M+1}(q-k_{M+1}+1)}{q} = \kappa_M. \tag{27}$$

Hence, by induction the hypothesis is true for all $0 \leq m \leq \mu$. In particular, we find that $P(\textsc{Tree}(x, a)) \geq \max(\epsilon_a - \kappa, 0)^K$. Now define the convex function,

$$f : \mathbb{R} \to \mathbb{R} : \quad x \mapsto \begin{cases} (x-\kappa)^K, & \text{if } x \geq \kappa, \\ 0, & \text{otherwise.} \end{cases} \tag{28}$$

Then the success probability of the extractor is at least

$$\mathbb{E}[f(\epsilon_a)] \geq f\left(\mathbb{E}[\epsilon_a]\right) = f(\epsilon(x)) = (\epsilon(x) - \kappa)^K, \tag{29}$$

where the first inequality follows by Jensen's inequality. This proves the theorem.

The following lemma generalizes Lemma 3 to interactive protocols in which the challenges sets are in the different rounds of the protocol are of different cardinalities.

Lemma 4. *Let $(\mathcal{P}, \mathcal{V})$ be a $(k_1, \dots, k_\mu)$-special sound $(2\mu+1)$-move interactive protocol for relation R, such that the verifier samples challenge c_i in move $2i$ uniformly at random from challenge set $\mathcal{C}_i$ for $1 \leq i \leq \mu$.*

Let x be some statement. Let $n_i := |\mathcal{C}_i|$ and let $\mathcal{P}^*$ be a prover such that $(\mathcal{P}^*, \mathcal{V})$ accepts with probability $\epsilon(x) > \kappa$, where

$$\kappa \leq \sum_{i=1}^{\mu} \frac{k_i - 1}{n_i}. \tag{30}$$

Then there exists a polynomial time extractor $\mathcal{E}$ with rewindable black-box access to $\mathcal{P}^*$ that on input x outputs a witness w for x with probability at least $(\epsilon(x) - \kappa)^K$ in at most K calls to $\mathcal{P}^*$, where $K = \prod_{i=1}^{\mu} k_i$.

The following lemma is a paraphrasing of Lemma 1 of [BCC+16] and it shows $(k_1, \dots, k_\mu)$-special soundness implies witness extended emulation.

Lemma 5 (Witness Extended Emulation [BCC+16]). *Let $(\mathcal{P}, \mathcal{V})$ be a $(k_1, \dots, k_\mu)$-special sound interactive protocol for relation R, where $\mathcal{V}$ samples each challenge uniformly at random from an exponentially sized challenge set. Suppose that $K = \prod_{i=1}^{\mu} k_i$ is polynomial in the security parameter. Then $(\mathcal{P}, \mathcal{V})$ has witness extended emulation.*

B Amortization over Many Commitments

In this section we describe how, from a standard amortization technique, a prover can show correctness of s evaluations of the linear form L on s different committed vectors for essentially the costs of evaluating one standard Σ-protocol. Compression then follows as before by which logarithmic communication complexity is achieved. We denote the corresponding relation by R^{Am}, i.e., every element of R^{Am} is composed of s elements of R. More precisely,

$$\begin{aligned} R^{\mathrm{Am}} = \Big\{ \big(P_1, \dots, P_s \in \mathbb{G}, L \in \mathcal{L}\left(\mathbb{Z}_q^n\right), y_1, \dots, y_s \in \mathbb{Z}_q; \\ \mathbf{x}_1, \dots, \mathbf{x}_s \in \mathbb{Z}_q^n, \gamma_1, \dots, \gamma_s \in \mathbb{Z}_q\big) : P_i = \mathbf{g}^{\mathbf{x}_i} h^{\gamma_i}, y_i = L(\mathbf{x}_i) \quad \forall i \Big\}. \end{aligned} \tag{31}$$

Note that the *same* linear form L is evaluated on different commitments.

Now note that for a uniform random challenge $c \in \mathbb{Z}_q$, the group element $\tilde{P} = A \prod_{i=1}^{s} P_i^{c^i}$ is a Pedersen commitment to $\tilde{\mathbf{x}} = \mathbf{r} + \sum_{i=1}^{s} \mathbf{x}_i c^i \in \mathbb{Z}_q^n$, where A is a commitment to a (random) vector $\mathbf{r}$. Moreover, it holds that $L(\tilde{\mathbf{x}}) = L(\mathbf{r}) + \sum_{i=1}^{s} y_i c^i$.

Furthermore, if there is a commitment P_j for which the prover does not know an opening then the prover knows an opening to $\tilde{P}$ with probability at most s/q. Informally, a cheating prover succeeds when c is the zero of some polynomial of degree at most s. When s is polynomial and q exponential in the security parameter this probability is exponentially close to 0.

The first two moves of the amortized Σ-protocol, denoted by Π_0^{Am}, are identical to that of Π_0. In the third move, the prover sends an opening to $\tilde{P}$, instead of an opening of AP^c, and the verifier performs the appropriate checks. Protocol Π_0^{Am} is $(s+1)$-special sound and knowledge soundness follows from Theorem 7 (Appendix A.1). The discussion is summarized in the following theorem.

Theorem 9 (Basic Pivot Amortized). *Π_0^{Am} is a 3-move protocol for relation R^{Am}. It is perfectly complete, special honest-verifier zero-knowledge and unconditionally $(s+1)$-special sound. Moreover, the communication costs are:*

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

The compressed Σ-protocol $\Pi_c^{\mathrm{Am}} := \Pi_2 \diamond \Pi_1 \diamond \Pi_0^{\mathrm{Am}}$ for relation R^{Am} achieves a logarithmic communication complexity. Its properties are summarized in the following theorem.

Theorem 10 (Compressed Pivot Amortized). *Π_c^{Am} is a $(2\mu+3)$-move protocol for relation R^{Am}, where $\mu = \lceil \log_2(n+1) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally $(s+1, 2, k_1, \dots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. Moreover, the communication costs are:*

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

C Compressed Pivot with Unconditional Soundness

In this section we describe two approaches to implement the compressed pivot with unconditional knowledge soundness, rather than computational.

First, observe that, in contrast to Bulletproofs [BBB+18], our compression mechanism *by itself* achieves unconditional soundness. The reason is that our pivot only considers linear constraints and no quadratic ones. The only building block that does not achieve unconditional soundness is the reduction of protocol Π_1. To achieve unconditional soundness, we simply omit this reduction. Only minor adaptations of the compression mechanism are required. A negative consequence of this approach is that the communication costs, while still logarithmic, are increased by a factor 2.

Second, we describe an approach that achieves unconditional soundness without incurring this factor 2 loss in communication efficiency. To this end, we show a conceptual simplification of Σ-protocol Π_0 that reduces the requirements for the compressed pivot to a minimum. Namely, to open linear forms, it turns out to be *sufficient* to have access to an efficient ZKPoK *for just opening Pedersen vector commitments.* Thus, in principle, a direct provision within the compressed pivot to show that a committed vector satisfies a linear constraint is no longer required.

We begin by showing that a ZK protocol for opening linear forms can be derived from a ZK protocol for *only* proving knowledge of an opening of a Pedersen vector commitment. More precisely, we show that a ZK protocol for relation

$$R' = \{(P \in \mathbb{G}; \mathbf{z} \in \mathbb{Z}_q^{n-1}, \gamma \in \mathbb{Z}_q) : \mathbf{k}^{\mathbf{z}} h^{\gamma} = P\}, \tag{32}$$

with public parameters $\mathbf{k} \in \mathbb{G}^{n-1}$ and $h \in \mathbb{G}$, implies a ZK protocol for relation R of Equation 1.

The main observation is that proving that a committed vector $\mathbf{x} \in \mathbb{Z}_q^n$ satisfies $L(\mathbf{x}) = y$, for some linear form L and scalar y, is equivalent to proving that $\mathbf{x}$ lies in the affine subspace $A_{L,y} = \{\mathbf{z} \in \mathbb{Z}_q^n : L(\mathbf{z}) = y\}$. We assume (w.l.o.g.) that $y = 0$ and that $L \neq 0$. Then $V_L := A_{L,0} \subset \mathbb{Z}_q^n$ is a linear subspace of dimension $n-1$. Both prover and verifier use the same deterministic algorithm to compute a basis $\mathbf{v}_1, \dots, \mathbf{v}_{n-1}$ for V_L and a new set of generators $\mathbf{k} := (\mathbf{g}^{\mathbf{v}_1}, \dots, \mathbf{g}^{\mathbf{v}_{n-1}}) \in \mathbb{G}^{n-1}$. Note that, since $\mathbf{v}_1, \dots, \mathbf{v}_{n-1}$ is a basis, a non-trivial discrete log relation between $k_1, \dots, k_{n-1}, h$ implies a non-trivial discrete log relation between $g_1, \dots, g_n, h$.

By black-box application of the ZK protocol for relation R' the prover shows that it knows an opening of commitment P with respect to generators $k_1, \dots, k_{n-1}, h$. From this it follows that the prover knows an opening $(\mathbf{x}, \gamma)$ of P with respect to generators $g_1, \dots, g_n, h$ such that $\mathbf{x} \in V_L$ and therefore $L(\mathbf{x}) = 0$, i.e., $(\mathbf{x}, \gamma)$ is a witness for relation R which completes the reduction.

Subsequently, we note that the simplified compressed pivot is immediately obtained from protocols Π_0 and Π_2 by "stripping them of their linear forms", i.e., all protocols steps involving the linear form L can be omitted. The reduction of protocol Π_1 is no longer required. As a result, the minimized compressed pivot described here achieves unconditional soundness.

Although this view may be superior from a conceptual standpoint, it does increase the computational costs for both the prover and the verifier. Both have to compute a basis for V_L and a new set of generators of $\mathbb{G}$. Moreover, at the point of writing, it is not clear how smoothly this approach carries over to assumptions other than the discrete log assumption. For these reasons, this paper is mainly based on the compressed pivot of Section 4.3.

D A Remark on Sublinear Communication Complexity

For his protocols Groth [Gro09] made the observation that there is a trade-off between the communication complexity and the number of rounds. A similar trade-off applies to our situation. Protocol Π_2 achieves a logarithmic communication complexity at the cost of a logarithmic number of rounds. The protocol recursively divides the witness into two parts, left and right. This idea is easily generalized to the situation in which the witness $\mathbf{x} \in \mathbb{Z}_q^n$ is divided into k parts.

For simplicity we assume n to be a power of k. A quick inspection of this generalization shows that instead of the two group elements A and B in the first round of Π_2, the prover has to sent $2k-2$ group elements. Recursing the protocol $\log_k(n)-1$ times results in a total communication of $(2k-2)\log_k(n)-2k+2$ elements of $\mathbb{G}$ and k elements of $\mathbb{Z}_q$ from prover to verifier. It is easily seen that these communication costs are minimized for $k=2$.

In contrast, $k=\sqrt{n/2}$ results in a constant round protocol with sublinear communication costs of $\sqrt{2n}-2$ elements of $\mathbb{G}$ and $\sqrt{2n}$ elements of $\mathbb{Z}_q$ from $\mathcal{P}$ to $\mathcal{V}$. Of course, in the non-interactive Fiat-Shamir mode the logarithmic variant might be preferable.

E Compactifying a Vector of Commitments

This section describes two solutions for the two extreme ZK cases, where the prover wishes to show that some relation holds for data that has already been committed to. Recall the two cases from Section 5.3:

– **Case 1:** The prover is committed to the input data $\mathbf{x}$ in a *single* compact commitment.
– **Case 2:** The prover is committed to the coordinates of the input data $\mathbf{x}$ *individually.*

Both solutions reduce the situation to that of a prover with a single compact commitment $[(\mathbf{x}, \mathsf{aux})]$ to all relevant data (i.e., input data and auxiliary data). Applying the methods of Section 6 *serially*, i.e., after these reductions, induces a factor 2 loss in the communication efficiency. Here, we show how these phases can be executed in *parallel*, amortizing their costs and thereby avoiding the factor 2 loss.

E.1 Case 1

The straightforward *serial* approach for this case is described in Section 5.3. It requires the prover to open one linear form on one compact commitment, and another linear form on another commitment. First, the amortized nullity checks on Pedersen commitment Q to $(0, \mathsf{aux}) \in \mathbb{Z}_q^{n+t}$ and, second, an opening on Pedersen commitment $P' = PQ$ to $(\mathbf{x}, \mathsf{aux}) \in \mathbb{Z}_q^{n+t}$ when applying the methods of Section 6. Recall that, here, P is a Pedersen commitment to the input data $\mathbf{x} \in \mathbb{Z}_q^n$. In this section we describe a Σ-protocol for precisely this amortization scenario where the linear form that is to be opened *depends* on the commitment.

More precisely, let us consider the two linear forms $L_1, L_2 : \mathbb{Z}_q^n \to \mathbb{Z}_q$ and two compact commitments $[\mathbf{x}_1]$ and $[\mathbf{x}_2]$ to $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{Z}_q^n$. The goal is to efficiently open $L_1(\mathbf{x}_1)$ and $L_2(\mathbf{x}_2)$ in ZK. In particular, the *cross-terms* $L_1(\mathbf{x}_2)$ and $L_2(\mathbf{x}_1)$ are to remain secret.

The main idea is to build a *shell* around the compact commitments that allows the prover to mask linear form evaluations that are not supposed to be revealed, i.e., the cross-terms. Thereby, the problem can be reduced to a standard amortization scenario where the entire "matrix" of linear form evaluations

$$\begin{pmatrix} L_1(\mathbf{x}_1) & L_1(\mathbf{x}_2) \\ L_2(\mathbf{x}_1) & L_2(\mathbf{x}_2) \end{pmatrix}$$

is revealed. More precisely, *intended* evaluations, on the diagonal of this matrix, will return the correct value and *unintended* evaluations will return a random, i.e., masked, value.

The solution presented here is constructed at the level of our basic Σ-protocol Π_0 (Section 3). Black-box access to $[\cdot]$ is insufficient and we therefore focus on the Pedersen vector commitment scheme. It suffices to consider a solution with linear communication complexity. A compressed version of this Σ-protocol, with logarithmic communication complexity, directly follows from the compression techniques of Section 4.

Let us now consider the details of our solution. Taking from the public set-up information a new pair of generators k_1, k_2 *disjoint* from the initial set that, supposedly, underlie Pedersen commitments $[\mathbf{x}_1]$ and $[\mathbf{x}_2]$. These additional generators allow the prover to incorporate additional (random) coefficients $u, w \in \mathbb{Z}_q$

in the commitments. However, it is essential that generator k_1 is only used to equip commitment $[\mathbf{x}_1]$ with a shell and generator k_2 is only used to equip commitment $[\mathbf{x}_2]$ with a shell. Shelled Pedersen commitments to $\mathbf{x}_1$ and $\mathbf{x}_2$ are obtained by multiplying $[\mathbf{x}_1]$ and $[\mathbf{x}_2]$ with shells k_1^u and k_2^w, respectively, and take the form $[(\mathbf{x}_1, u, 0)]$ and $[(\mathbf{x}_2, 0, w)]$.

Altogether we have reduced the problem to finding a ZK protocol for the following relation,

$$\begin{aligned} R_{\text{shell}} = \{\, & (P_1, P_2, L_1, L_2, y_1, y_2; \mathbf{x}_1, \mathbf{x}_2, u, w, \gamma_1, \gamma_2) : \\ & P_1 = \mathbf{g}^{\mathbf{x}_1} k_1^u h^{\gamma_1}, P_2 = \mathbf{g}^{\mathbf{x}_2} k_2^w h^{\gamma_2}, \\ & L_1(\mathbf{x}_1) = y_1, L_2(\mathbf{x}_2) = y_2 \}, \end{aligned} \tag{33}$$

where $\mathbf{g} \in \mathbb{G}^n$ and $k_1, k_2, h \in \mathbb{G}$ are public parameters, $P_1, P_2 \in \mathbb{G}$, $u, w, y_1, y_2, \gamma_1, \gamma_2 \in \mathbb{Z}_q$, $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{Z}_q^n$ and $L_1, L_2 : \mathbb{Z}_q^n \to \mathbb{Z}_q$ are linear forms.

Next, we describe how this relation can be reduced to the standard amortization scenario where cross terms *are* revealed. Simultaneously, we introduce amortized nullity checks on the appropriated coordinates of $(\mathbf{x}_1, u, 0)$ and $(\mathbf{x}_2, 0, w)$. These nullity checks have to be incorporated at this stage of the protocol, otherwise they will again introduce cross terms that are to remain secret. So let $\rho \in \mathbb{Z}_q \setminus \{-1\}$ be a uniform random challenge and let us consider the following linear forms:

$$\begin{aligned} \widehat{L}_1 &: \mathbb{Z}_q^{n+2} \to \mathbb{Z}_q, \quad (\mathbf{x}, a, b) \mapsto L_1(\mathbf{x}) - y_1 + b(\rho + 1), \\ \widehat{L}_2 &: \mathbb{Z}_q^{n+2} \to \mathbb{Z}_q, \quad (\mathbf{x}, a, b) \mapsto L_2(\mathbf{x}) - y_2 + a(\rho + 1). \end{aligned} \tag{34}$$

Note that to mask the cross terms it is essential that $\rho \neq -1$.

From these derived linear forms the following relation, with public cross-terms, arises:

$$\begin{aligned} \widehat{R}_{\text{shell}} = \{\, & (\mathbf{g}, k_1, k_2, h, P_1, P_2, L_1, L_2, y_1, y_2, y_{11}, y_{21}; \mathbf{x}_1, \mathbf{x}_2, u, w, u', w', \gamma_1, \gamma_2) : \\ & P_1 = \mathbf{g}^{\mathbf{x}_1} k_1^u k_2^{w'} h^{\gamma_1}, P_2 = \mathbf{g}^{\mathbf{x}_2} k_1^{u'} k_2^w h^{\gamma_2}, \widehat{L}_1(\mathbf{x}_1, u, w') = \widehat{L}_2(\mathbf{x}_2, u', w) = 0, \\ & \widehat{L}_1(\mathbf{x}_2, u', w) = y_{12}, \widehat{L}_2(\mathbf{x}_1, u, w') = y_{21} \}. \end{aligned} \tag{35}$$

Standard amortization techniques directly yield a Σ-protocol for $\widehat{R}_{\text{shell}}$ with the desired communication complexity. However, a ZKPoK for relation $\widehat{R}_{\text{shell}}$ does not yield a ZKPoK for relation R_{shell}, since information about the masks u and w is revealed. For this reason, the prover first re-randomizes the shells by sending commitments R_1, R_2 to $s_1, s_2 \in \mathbb{Z}_q$ chosen uniformly at random under generators (k_1, h) and (k_2, h), respectively. The prover and verifier compute the re-randomized commitments $R_1 P_1$ and $R_2 P_2$ and by two standard Σ-protocols the prover shows that commitments R_1 and R_2 exclusively involve the appropriate generators. After re-randomization the prover and verifier run a standard amortized Σ-protocol for relation $\widehat{R}_{\text{shell}}$.

To summarize, the ZKPoK for relation R_{shell} contains three components:

1. Amortized nullity checks on shelled commitments P_1 and P_2.
2. Re-randomization of the shells, together with the basic Σ-protocols for R_1 and R_2.
3. Amortized Σ-protocol for relation $\widehat{R}_{\text{shell}}$.

The protocol is formally described in Protocol 9 and denoted by Π_{shell}. Theorem 11 summarizes its main properties. To achieve logarithmic communication complexity, compression can be applied as before. Moreover, we note that a straightforward generalization allows the compactification of any $s > 1$ compact commitments.

Theorem 11. *Π_{shell} is a 4-move protocol for relation R_{shell}. It is perfectly complete, special honest verifier zero-knowledge and computationally $(2, 3)$-special sound, under the discrete logarithm assumption. Moreover, the communication costs are:*

– *$\mathcal{P} \to \mathcal{V}$: 5 elements of $\mathbb{G}$ and $n + 11$ elements of $\mathbb{Z}_q$.*

– $\mathcal{V} \to \mathcal{P}$: 2 *elements of* $\mathbb{Z}_q$.

Proof. **Completeness** directly follows.

SHVZK: On input $\rho, c \in \mathbb{Z}_q$ with $\rho \neq -1$, the simulator samples $R_1, R_2 \in \mathbb{G}, y_{21}, y_{12}, z_1, z_2, \phi_1, \phi_2, \phi \in \mathbb{Z}_q$ and $\mathbf{z} \in \mathbb{Z}_q^{n+2}$ uniformly at random. From these values the simulator computes:

$$\begin{aligned} A &= (\mathbf{g}, k_1, k_2)^{\mathbf{z}} h^{\phi} (P_1 R_1)^{-c} (P_2 R_2)^{-c^2}, \\ A_1 &= k_1^{z_1} h^{\phi_1} R_1^{-c}, \\ A_2 &= k_2^{z_2} h^{\phi_2} R_2^{-c}, \\ t_1 &= \widehat{L}_1(\mathbf{z}) - c^2 y_{12}, \\ t_2 &= \widehat{L}_2(\mathbf{z}) - c y_{21}. \end{aligned} \tag{36}$$

The resulting transcript tr is now easily seen to be accepting. Moreover, using the fact that $\rho \neq -1$, it follows that the probability distribution of tr is equal to that of an honest execution.

Special Soundness: Let us now show that Π_{shell} is $(2,3)$-special sound. From three accepting transcripts with coinciding first and second messages and different challenges, the extractor uses standard techniques to extract elements $\bar{\mathbf{z}}_1, \bar{\mathbf{z}}_2 \in \mathbb{Z}_q^{n+2}$ and $\bar{\phi}_1, \bar{\phi}_2, \bar{z}_1, \bar{z}_2, \bar{\phi}_{11}, \bar{\phi}_{21} \in \mathbb{Z}_q$ for which it holds that

$$\begin{aligned} &(\mathbf{g}, k_1, k_2)^{\bar{\mathbf{z}}_1} h^{\bar{\phi}_1} = P_1 R_1, && \widehat{L}_1(\bar{\mathbf{z}}_1) = 0, && k_1^{\bar{z}_1} h^{\bar{\phi}_{11}} = R_1, \\ &(\mathbf{g}, k_1, k_2)^{\bar{\mathbf{z}}_2} h^{\bar{\phi}_2} = P_2 R_2, && \widehat{L}_2(\bar{\mathbf{z}}_2) = 0, && k_2^{\bar{z}_2} h^{\bar{\phi}_{21}} = R_2. \end{aligned} \tag{37}$$

Now let us define $\widehat{\mathbf{z}}_1 := \bar{\mathbf{z}}_1 - (0, \dots, 0, \bar{z}_1, 0)$, $\widehat{\mathbf{z}}_1 := \bar{\mathbf{z}}_2 - (0, \dots, 0, \bar{z}_2)$, $\widehat{\phi_1} := \bar{\phi}_1 - \bar{\phi}_{11}$ and $\widehat{\phi_2} := \bar{\phi}_2 - \bar{\phi}_{21}$. Then it holds that

$$\begin{aligned} &(\mathbf{g}, k_1, k_2)^{\widehat{\mathbf{z}}_1} h^{\widehat{\phi}_1} = P_1, && \widehat{L}_1(\widehat{\mathbf{z}}_1) = 0, \\ &(\mathbf{g}, k_1, k_2)^{\widehat{\mathbf{z}}_2} h^{\widehat{\phi}_2} = P_2, && \widehat{L}_2(\widehat{\mathbf{z}}_2) = 0. \end{aligned} \tag{38}$$

The remainder of the proof follows from an analysis of the polynomial amortization trick. Namely, rewinding once more and running the same procedure for a different first move challenge ρ' allows the extraction of another set of elements $\widetilde{\mathbf{z}}_1, \widetilde{\mathbf{z}}_2, \widetilde{\phi}_1, \widetilde{\phi}_2$ and an affine form $\widetilde{L}$ satisfying the relations of Equation 38. Hence, either the extractor has found a non-trivial discrete log relation or $\widetilde{\mathbf{z}}_1 = \widehat{\mathbf{z}}_1$, $\widetilde{\mathbf{z}}_2 = \widehat{\mathbf{z}}_2$, $\widetilde{\phi}_1 = \widehat{\phi}_1$ and $\widetilde{\phi}_2 = \widehat{\phi}_2$. In the latter case it follows that $\widehat{\mathbf{z}}_1 = (\widehat{\mathbf{x}}_1, a_1, b_1)$ and $\widehat{\mathbf{z}}_2 = (\widehat{\mathbf{x}}_2, a_2, b_2)$ satisfy $b_1 = a_2 = 0$ and

$$\begin{aligned} &\mathbf{g}^{\widehat{\mathbf{x}}_1} k_1^{a_1} h^{\widehat{\phi}_1} = P_1, && L_1(\widehat{\mathbf{x}}_1) = 0, \\ &\mathbf{g}^{\widehat{\mathbf{x}}_2} k_2^{b_2} h^{\widehat{\phi}_2} = P_2, && L_2(\widehat{\mathbf{x}}_2) = 0. \end{aligned} \tag{39}$$

Hence, Protocol Π_{shell} is $(2,3)$-special sound which proves the Theorem.

E.2 Case 2

Let us consider the case where the prover has s individual Pedersen commitments P_i to $v_i \in \mathbb{Z}_q$. The goal is to construct an interactive protocol that takes as public input $P_1, \dots, P_s$, as prover's private input $v_1, \dots, v_s$ and $\mathsf{aux} \in \mathbb{Z}_q^t$ and outputs a compact commitment to $[(v_1, \dots, v_s, \mathsf{aux})]$. In fact, our solution outputs a commitment $[(v_1, \dots, v_s, r, \mathsf{aux})]$ where $r \in \mathbb{Z}_q$ is sampled uniformly at random. This additional coefficient r does not introduce any difficulties in applying the methods of Section 6. In contrast to Case 1, we will see that parallelization with this compactification technique immediately follows. Moreover, black-box access to our compressed pivot $[\cdot]$ suffices.

The main idea is to create a new compact commitment to $(v_1, \dots, v_s, r, \mathsf{aux}) \in \mathbb{Z}_q^{s+t+1}$ and use a standard (amortized) Σ-protocol to prove correctness of this compact commitment. The amortized Σ-protocol was, for the more general Pedersen vector commitments, described in Appendix B. More precisely, from the

Protocol 9 Zero-Knowledge Proof of Knowledge Π_{shell} for relation R_{shell}
Amortizing the costs of opening two different linear forms on two different Pedersen vector commitments.

```
                    PUBLIC PARAMETERS : $\mathbf{g}, k_1, k_2, h$
    INPUT$(P_1, P_2, L_1, L_2, y_1, y_2; \mathbf{x}_1, \mathbf{x}_2, u, w, \gamma_1, \gamma_2)$
                    $P_1 = \mathbf{g}^{\mathbf{x}_1} k_1^u h^{\gamma_1} \in \mathbb{G}$
                    $P_2 = \mathbf{g}^{\mathbf{x}_2} k_2^w h^{\gamma_2} \in \mathbb{G}$
                    $y_1 = L_1(\mathbf{x}_1), y_2 = L_2(\mathbf{x}_2)$
Prover                                                              Verifier
                    $\xleftarrow{\rho}$                              $\rho \leftarrow_R \mathbb{Z}_q \setminus \{-1\}$
                    $\widehat{L}_1(\mathbf{x}, a, b) := L_1(\mathbf{x}) - y_1 + b(\rho + 1)$
                    $\widehat{L}_2(\mathbf{x}, a, b) := L_2(\mathbf{x}) - y_2 + a(\rho + 1)$
$s_1, s_2, \psi_1, \psi_2 \leftarrow_R \mathbb{Z}_q$
$R_1 = k_1^{s_1} h^{\psi_1}, R_2 = k_2^{s_2} h^{\psi_2}$

$y_{21} = \widehat{L}_2(\mathbf{x}_1, u + s_1, 0)$
$y_{12} = \widehat{L}_1(\mathbf{x}_2, 0, w + s_2)$

$r_1, r_2, \eta_1, \eta_2 \leftarrow \mathbb{Z}_q$
$A_1 = k_1^{r_1} h^{\eta_1}, A_2 = k_2^{r_2} h^{\eta_2}$

$\mathbf{r} \leftarrow_R \mathbb{Z}_q^{n+2}, \omega \leftarrow_R \mathbb{Z}_q$
$t_1 = \widehat{L}_1(\mathbf{r}), t_2 = \widehat{L}_2(\mathbf{r})$
$A = (\mathbf{g}, k_1, k_2)^{\mathbf{r}} h^{\omega}$
                    $\xrightarrow{R_1, R_2, A, A_1, A_2, y_{21}, y_{12}, t_1, t_2}$
                                                                    $c \leftarrow_R \mathbb{Z}_q$
                    $\xleftarrow{c}$
$\mathbf{z} = (\mathbf{x}_1, u + s_1, 0)c +$
$(\mathbf{x}_2, 0, w + s_2)c^2 + \mathbf{r}$
$\phi = (\gamma_1 + \psi_1)c + (\gamma_2 + \psi_2)c^2 + \omega$
$\tilde{z}_1 = cs_1 + r_1$
$\phi_1 = c\psi_1 + +\eta_1$
$\tilde{z}_2 = cs_2 + r_2$
$\phi_2 = c\psi_2 + \eta_2$
                    $\xrightarrow{\mathbf{z}, \phi, \tilde{z}_1, \phi_1, \tilde{z}_2, \phi_2}$
                                                    $(\mathbf{g}, k_1, k_2)^{\mathbf{z}} h^{\phi} \stackrel{?}{=} A(P_1 R_1)^c (P_2 R_2)^{c^2}$
                                                    $\widehat{L}_1(\mathbf{z}) \stackrel{?}{=} c^2 y_{12} + t_1$
                                                    $\widehat{L}_2(\mathbf{z}) \stackrel{?}{=} c y_{21} + t_2$
                                                    $k_1^{\tilde{z}_1} h^{\phi_1} \stackrel{?}{=} A_1 R_1^c$
                                                    $k_2^{\tilde{z}_2} h^{\phi_2} \stackrel{?}{=} A_2 R_2^c$
```

amortized Σ-protocol for proving knowledge of openings to $P_1, \ldots, P_s$, we construct a new protocol Π_P. The first message of the basic Σ-protocol is appended with a compact commitment $[\mathbf{y}] = [(v_1, \ldots, v_s, r, \mathsf{aux})]$, where r is the random element to which the prover committed in the first round of the Σ-protocol. After the final round of the Σ-protocol, the prover and verifier run the interactive nullity check on compact commitment $[\mathbf{y}]$ and affine form $L_c(\mathbf{x}) - z = x_{s+1} + \sum_{i=1}^{s} c^i x_i - z$, where c is the verifier's challenge and z is the prover's response. This nullity check shows that the commitment $[\mathbf{y}]$ is of the correct form.

Note that it is immaterial that P is a Pedersen commitment. Any other commitment scheme with a Σ-protocol that satisfies some linearity constraints suffices. In particular, the thirds message should be computed as the evaluation of a public linear form parameterized by the challenge c. Moreover, because Π_P only performs a nullity check on compact commitment $[\mathbf{y}]$, parallelization with the methods of Section 6 directly follows. Namely, these methods follow from partial openings of exactly the same vector $[\mathbf{y}]$.

The protocol is formally described in Protocol 10. It outputs a vector commitment $[\mathbf{y}]$ and is a ZK protocol for the following relation

$$R_P = \{(P_1, \dots, P_s, [\mathbf{y}]; v_1, \gamma_1, \dots v_s, \gamma_s, \mathbf{y}) : P_j = g^{v_j} h^{\gamma_j}, v_j = y_j \quad 1 \le j \le s\}\,. \tag{40}$$

The properties of Π_P are summarized in Theorem 12.

Theorem 12. *Π_P is a $(2\mu + 5)$-move protocol for relation R_P, where $\mu = \lceil \log_2(s + t + 2) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally $(s + 1, 2, 2, k_1, \dots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \le i \le \mu$. Moreover, the communication costs are:*

- $\mathcal{P} \to \mathcal{V}$: $2 \lceil \log_2(s + t + 2) \rceil + 1$ *elements of* $\mathbb{G}$ *and* 5 *elements of* $\mathbb{Z}_q$.
- $\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(s + t + 2) \rceil + 2$ *elements of* $\mathbb{Z}_q$.

Proof. **Completeness** and **SHVZK** follow from the associated properties of the Σ-protocol and Π_{NULLITY}.

Special soundness: We first recall the proof that shows that the basic amortized Σ-protocol is $(s+1)$-special sound. Let $(A, c_j, \mathbf{z}_j, \phi_i)$ be accepting transcripts for $0 \le j \le s$ with $c_i \ne c_j$ for all $i \ne j$. Let V be the $(s+1) \times (s+1)$ Vandermonde matrix generated by $c_0, \dots, c_s$. Since the challenges are distinct, it follows that V is invertible. Let $V^{-1} = (a_{i,j})_{0 \le i,j \le s}$ and, for $1 \le \ell \le s$, let

$$\begin{aligned} \bar{z}_\ell &:= \sum_{j=0}^{s} a_{\ell,j} \mathbf{z}_j, \\ \bar{\gamma}_\ell &:= \sum_{j=0}^{s} a_{\ell,j} \phi_j. \end{aligned} \tag{41}$$

Then, it holds that $g^{\bar{z}_\ell} h^{\bar{\gamma}_\ell} = P_\ell$ for all ℓ, which proves that the basic amortized Σ-protocol is $(s+1)$-special sound.

Special soundness of the compound protocol now follows by the $(2, 2, k_1, \dots, k_\mu)$-special soundness of Π_{NULLITY} and the fact that for all ℓ

$$\sum_{j=0}^{s} a_{\ell,j} L_{c_j}(\mathbf{x}) = x_\ell. \tag{42}$$

F Range Proof

In this section we show how two variations of range proofs can be derived as an immediate consequence of the circuit ZK protocols of Section 6. First, we treat the basic scenario where a prover wishes to commit to a secret integer v and show that this integer is in a public range, say $[0, 2^{n-1}]$. Second, we consider the scenario where the prover wishes to convince a verifier that many different integer commitments are all in some fixed range.

F.1 Basic Range Proofs

In the basic scenario the prover commits to the integer $v \in \{0, \dots, 2^{n-1}\}$ and the required auxiliary data aux at once in a single compact commitment. In the following solution the prover does not commit to the integer v explicitly, but only via its bit-decomposition $\mathbf{b} \in \mathbb{Z}_q^n$. Namely, note that v can be computed as linear form

Protocol 10 Extended Σ-protocol Π_P for s Pedersen commitments

$$\textsc{Input}(P_1,\dots,P_s;v_1,\gamma_1,\dots v_s,\gamma_s,\mathsf{aux})$$
$$\textsc{Output}([v_1,\dots,v_s,r,\mathsf{aux}])$$

$$P_j = g^{v_j}h^{\gamma_j}$$
$$L_a(\mathbf{x}) := x_{s+1} + \sum_{i=1}^{s} a^i x_i$$

Prover		Verifier
$r \leftarrow_R \mathbb{Z}_q, \rho \leftarrow_R \mathbb{Z}_q$	
$A = g^r h^\rho$	
$\mathbf{y} = (v_1,\dots,v_s,r,\mathsf{aux})$	
	$\xrightarrow{A,[\mathbf{y}]}$
		$c \leftarrow_R \mathbb{Z}_q$
	$\xleftarrow{c}$
$z = r + \sum_{i=1}^{s} c^i v_i$	
$\phi = \rho + \sum_{i=1}^{s} c^i \gamma_i$	
	$\xrightarrow{z,\phi}$
		$g^z h^\phi \overset{?}{=} A\prod_{i=1}^{s} P^{c^i}$

$$\Pi_{\textsc{Nullity}}([\mathbf{y}], L_c - z; \mathbf{y})$$

evaluated at $\mathbf{b}$, hence a compact commitment to $\mathbf{b}$ is an *implicit* commitment to v. To show that v is in the range $[0, 2^{n-1}]$, the prover now only has to convince the verifier that the committed vector $\mathbf{b}$ is comprised of 0's and 1's, which can be done by a simple application of the circuit ZK protocol Π_{cs}.

To this end, let $C : \mathbb{Z}_q^n \to \mathbb{Z}_q^n, \mathbf{x} \mapsto \mathbf{x} * (1 - \mathbf{x})$. Prover and verifier run Π_{cs} on input $(C; \mathbf{b})$ to obtain a ZK protocol for relation

$$R_r = \{(C; \mathbf{b}) : C(\mathbf{b}) = 0\}\,. \tag{43}$$

Minor improvements to the protocol are obtained by observing that:

1. All multiplication gates have inputs of the form α and $1 - \alpha$. Hence, instead of sampling a random polynomial $g(X)$ for the right inputs of multiplication gates we take $g(X) = 1 - f(X)$.
2. All outputs of multiplications gates are 0, hence $h(1) = h(2) = \cdots = h(n) = 0$ and these values do not have to be included in the compact commitment.

The full protocol, denoted by Π_r, is described in Protocol 11. Theorem 13 shows that Π_r is a SHVZK argument of knowledge for relation R_r.

Theorem 13 (Basic Range Proof). *Π_r is a $(2\mu + 7)$-move protocol for relation R_r, where $\mu = \lceil \log_2(2n + 3)\rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally $(2n + 1, n + 2, 2, 2, k_1, \dots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \le i \le \mu$. Moreover, the communication costs are:*

- *$\mathcal{P} \to \mathcal{V}$: $2\lceil \log_2(2n + 3)\rceil$ elements of $\mathbb{G}$ and 5 elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(2n + 3)\rceil + 3$ elements of $\mathbb{Z}_q$.*

F.2 Compactifying Many Pedersen Commitments

Let us now consider the case of a prover that wishes to show that s Pedersen commitments to $v_1, \dots, v_s \in \mathbb{Z}_q$ are all in the range $[0, 2^{n-1}]$. For the corresponding relation we write $R_r^{(s)}$. The protocol, denoted by $\Pi_r^{(s)}$, is constructed by deploying a minor adaptation of the "Case 2" compactification techniques of Section 5.3. The compactification techniques are easily adapted to obtain a single compact commitment to the ns bits of the s committed values together with the auxiliary data required to prove their correctness. The properties of $\Pi_r^{(s)}$ are given by the following theorem.

Protocol 11 Range proof Π_r

The polynomial f is sampled uniformly at random such that its evaluations at $1, \ldots, n$ correspond to $\mathbf{b}$.

$$\textsc{Input}(C; \mathbf{b})$$

$$C : \mathbb{Z}_q^n \to \mathbb{Z}_q^n, \quad \mathbf{x} \mapsto \mathbf{x} * (1 - \mathbf{x}), \quad C(\mathbf{b}) = 0$$

$$\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
f \leftarrow_R \mathbb{Z}_q[X]_{\leq n} & & \\
h(X) = f(X)(1 - f(X)) & & \\
\mathbf{y} = (\mathbf{b}, f(0), h(0), h(n+1), \ldots, h(2n)) & \xrightarrow{[\mathbf{y}]} & \\
 & & c \leftarrow_R \mathbb{Z}_q \setminus \{1, \ldots, n\} \\
 & \xleftarrow{c} & \\
z_1 = f(c) & & \\
z_2 = h(c) & & \\
 & \xrightarrow{z_1, z_2} & \\
 & & z_2 \overset{?}{=} z_1(1 - z_1)
\end{array}$$

$$\Pi_{\textsc{Nullity}} \begin{pmatrix} C(\mathbf{b}) \\ [\mathbf{y}], f(c) - z_1; \mathbf{y} \\ h(c) - z_2 \end{pmatrix}$$

Theorem 14 (Range Proof Case 2). *$\Pi_r^{(s)}$ is a $(2\mu + 7)$-move protocol for relation $R_r^{(s)}$, where $\mu = \lceil \log_2(2ns + 4) \rceil - 1$. It is perfectly complete, special honest-verifier zero-knowledge and computationally $(s + 1, 2ns + 1, ns + 3, 2, 2, k_1, \ldots, k_\mu)$-special sound, under the discrete logarithm assumption, where $k_i = 3$ for all $1 \leq i \leq \mu$. Moreover, the communication costs are:*

- *$\mathcal{P} \to \mathcal{V}$: $2 \lceil \log_2(2ns + s + 4) \rceil + 1$ elements of $\mathbb{G}$ and 7 elements of $\mathbb{Z}_q$.*
- *$\mathcal{V} \to \mathcal{P}$: $\lceil \log_2(2ns + s + 4) \rceil + 4$ elements of $\mathbb{Z}_q$.*

G Strong-RSA Assumption

In this appendix we informally sketch the approach of [BFS20] along with our adaptations to allow for the opening of *arbitrary* linear forms. This adaptations can be used to base our pivot on assumptions derived from the Strong-RSA assumption.

G.1 Integer Commitment Scheme

We briefly recall the integer commitment scheme of [DF02]. The commitment space of this scheme is a group $\mathbb{G}$ of unknown order, such as an RSA group or a class group. Although the exact order of $\mathbb{G}$ is unknown, we do assume to know an upper bound B on the order, i.e., $|\mathbb{G}| \leq B$.

The setup phase of the commitment scheme generates two random group elements $g, h \in \mathbb{G}$ such that they both generate the same subgroup of $\mathbb{G}$. In this case the distribution of h^γ for γ chosen uniformly at random from $[0, B \cdot 2^\kappa)$, where κ is the security parameter, will be exponentially close to the uniform distribution on $\langle g \rangle$. Hence for an arbitrary integer x, the element $[x] = g^x h^\gamma \in \mathbb{G}$ statistically hides x.

Intuitively, the binding property follows from the assumption that the prover does not know the order of $\mathbb{G}$. Formally, the binding property can be shown to follow from the root assumption [DF02, BFS20].

G.2 Vector Encoding

The vector encoding scheme of [BFS20] first lifts vectors $\mathbf{x} \in \mathbb{Z}_q^n$ to their unique representatives in $\mathbb{Z}\left(\frac{q-1}{2}\right)^n = \left\{\mathbf{x} \in \mathbb{Z}^n : \|\mathbf{x}\|_\infty \le \frac{q-1}{2}\right\}$. Subsequently, for any $b \in \mathbb{Z}$ and $Q > 2b$ the following encoding is applied:

$$\text{Encode} : \mathbb{Z}(b)^n \to \mathbb{Z}, \quad \mathbf{x} \mapsto \sum_{i=1}^{n} x_i Q^{i-1}. \tag{44}$$

This encoding is injective since $Q > 2b$. For both $\mathbf{x} \in \mathbb{Z}_q^n$ and $\mathbf{x} \in \mathbb{Z}(b)^n$, we will write $\widehat{\mathbf{x}} \in \mathbb{Z}$ for their integer encodings. A commitment $[\mathbf{x}]$ to a vector $\mathbf{x} \in \mathbb{Z}_q^n$ or $\mathbf{x} \in \mathbb{Z}(b)^n$ is an integer commitment to $\widehat{\mathbf{x}}$.

G.3 *Σ*-Protocol

The above thus generates a compact vector commitment scheme $[\cdot] : \mathbb{Z}_q^n \to \mathbb{G}$. For a linear form $L : \mathbb{Z}_q^n \to \mathbb{Z}_q$, this commitment scheme has a basic Σ-protocol for the relation

$$R_{\mathbb{Z}_q} = \left\{\left(P \in \mathbb{G}, u \in \mathbb{Z}_q, Q \in \mathbb{Z}, L; \mathbf{x} \in \mathbb{Z}_q^n, \gamma \in \mathbb{Z}_q\right) : P = g^{\widehat{\mathbf{x}}} h^\gamma, L(\mathbf{x}) = \mathbf{u}, Q > q\right\}. \tag{45}$$

The main differences between this Σ-protocol and protocol Π_0 from Section 3 is that the protocol is statistically hiding and all exponents are sampled from subsets of $\mathbb{Z}$. For this reason, the verifier has to check that the final response is of bounded norm. A similar Σ-protocol is described in [BFS20].

Protocol 12 Basic Σ-protocol for inner product relation $R_{\mathbb{Z}_q}$

Public Parameters : g, h
Input$(P, Q, L; \mathbf{x}, \gamma)$

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

$$\begin{array}{ccc}
\text{Prover} & & \text{Verifier} \\
\mathbf{r} \leftarrow_R \mathbb{Z}\left((q-1)^2 2^{\kappa-2}\right)^n & & \\
\rho \leftarrow_R [0, B \cdot 2^\kappa) & & \\
t = L(\mathbf{r}) \mod q & & \\
A = g^{\widehat{\mathbf{r}}} h^\rho & & \\
& \xrightarrow{t, A} & \\
& & c \leftarrow_R \left[-\frac{q-1}{2}, \frac{q-1}{2}\right] \\
& \xleftarrow{c} & \\
\mathbf{z} = c\mathbf{x} + \mathbf{r} \in \mathbb{Z}^n & & \\
\phi = c\gamma + \rho \in \mathbb{Z} & & \\
& \xrightarrow{\phi, \mathbf{z}} & \\
& & g^{\mathbf{z}} h^\phi \overset{?}{=} P^c A \\
& & \|\mathbf{z}\|_\infty \overset{?}{\le} q^2 2^{\kappa-1} \\
& & L(\mathbf{z}) \overset{?}{=} cu + t
\end{array}$$

G.4 Compressed *Σ*-Protocol

The protocol can be compressed by observing that the response $\mathbf{z}$ is, in fact, a trivial PoK for the relation $R_{\mathbb{Z}}$.

$$R_{\mathbb{Z}} = \left\{\left(P \in \mathbb{G}, u \in \mathbb{Z}_q, Q, b \in \mathbb{Z}, L; \mathbf{x} \in \mathbb{Z}^n\right) : \|\mathbf{x}\|_\infty \le b < q, P = g^{\widehat{\mathbf{x}}}, L(\mathbf{x}) = u \mod p\right\}. \tag{46}$$

Following Bulletproof's recursive techniques a more efficient PoK for relation $R_{\mathbb{Z}}$ can be constructed. Protocol 13 shows one iteration of the recursion, repeating this recursion $O(\log n)$ times results in a logarithmic complexity. It must be noted that the bound b grows in each iteration. For this reason the encoding parameter Q has to be chosen large enough. The polynomial evaluation protocol of [BFS20] replaces the computationally expensive exponentiation after the first move (i.e., computing $A_R^{n/2}$) by a Proof of Exponentiation [Wes19], thereby reducing the verification time. For details we refer to [BFS20].

Another difference between this approach and the compression in the discrete log setting is that here the linear form evaluation $L(\mathbf{x})$ is not included in the commitment. For this reason the cross terms A_R and A_L are part of the first message.

Protocol 13 Compressed Argument of Knowledge for relation $R_{\mathbb{Z}}$

$$\textsc{Public Parameters} : g$$
$$\textsc{Input}(P, u, Q, b, L; \mathbf{x})$$

$$\mathbf{x} \in \mathbb{Z}(b)^n$$
$$P = g^{\widehat{\mathbf{x}}}$$
$$L(\mathbf{x}) = u \mod q$$

$$\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
A_L \leftarrow g^{\widehat{\mathbf{x}}_L} & & \\
A_R \leftarrow g^{\widehat{\mathbf{x}}_R} & & \\
u_L = L_R(\mathbf{x}_L) & & \\
u_R = L_L(\mathbf{x}_R) & & \\
 & \xrightarrow{A_L, A_R \mathbf{u}_L, \mathbf{u}_R} & \\
 & & A_L A_R^{n/2} \stackrel{?}{=} P \\
 & & c \leftarrow_R \left\{-\frac{p-1}{2}, \dots, \frac{p-1}{2}\right\} \\
 & \xleftarrow{c} & \\
\mathbf{z} = c\mathbf{x}_L + \mathbf{x}_R & & \\
 & \xrightarrow{\mathbf{z}} & g^{\widehat{\mathbf{z}}} \stackrel{?}{=} A_L^c A_R \\
 & & (L_L + cL_R)(\mathbf{z}) \stackrel{?}{=} \\
 & & cu + c^2 u_L + u_R \\
 & & 0 \stackrel{?}{\leq} b \stackrel{?}{<} \frac{Q}{2} \\
 & & \|\mathbf{z}\|_\infty \stackrel{?}{<} b\frac{q+1}{2}
\end{array}$$