A Zero-Knowledge Version of vSQL

Yupeng Zhang
University of Maryland
zhangyp@umd.edu

Daniel Genkin
University of Pennsylvania and
University of Maryland
danielg3@cis.upenn.edu

Jonathan Katz
University of Maryland
jkatz@cs.umd.edu

Dimitrios Papadopoulos
Hong Kong University of Science and Technology
dipapado@cse.ust.hk

Charalampos Papamanthou
University of Maryland
cpap@umd.edu

Abstract

Zero-knowledge arguments of knowledge are powerful cryptographic primitives that allow a computationally strong prover to convince a weaker verifier for the validity of an NP statement, without revealing anything about the corresponding witness (beyond its existence). Most state-of-the art implementations of such arguments that achieve succinct communication and verification cost follow the quadratic arithmetic program paradigm. One notable exception to this is the vSQL system of [Zhang et al. IEEE S&P 2017] which takes an entirely different approach resulting is significantly fewer cryptographic operations. However, it has the notable downside that is not zero-knowledge (i.e., it does not hide the witness from the verifier), a property that has proven to be of utmost importance in many application (e.g., in cryptocurrencies). In this work, we present a zero-knowledge version of the argument upon which vSQL is based. Our construction utilizes two separate techniques: (i) a novel zero-knowledge verifiable polynomial delegation protocol, and (ii) running parts of the argument of vSQL over homomorphic commitments, thus hiding the committed values.

1 Introduction

Protocols for *verifiable computation (VC)* allow a computationally weak verifier to outsource the execution of a program to a powerful but untrusted prover (e.g., a cloud provider) while being assured that the result was computed correctly. Somewhat more formally, a verifier $\mathcal{V}$ and prover $\mathcal{P}$ agree on a circuit C and an input x. The prover then sends a result y to the verifier, and proves that there exists some w for which $y = C(x; w)$. (Here, w represents some auxiliary input chosen by the prover.) Starting with the work of Kilian [23], there has been a long line of work constructing VC protocols for arbitrary computations. For VC protocols that support general programs, the most prominent such works rely on succinct arguments-of-knowledge (SNARKs) [5] based on quadratic arithmetic programs [17]. This has resulted in several optimized systems that achieve excellent performance. See [29] and followup works. Most constructions use a preprocessing phase during which a trusted party (possibly the verifier itself) generates a set of public parameters corresponding to the specific circuit C of interest. This can be used in two ways:

1. **Computation-specific circuit.** If the verifier knows ahead of time the circuit C he wants to evaluate, he can simply execute the preprocessing phase for C. Numerous works follow this approach, e.g., [26, 13].

2. **Universal circuit.** If the computation to be verified cannot be determined during preprocessing, the verifier can always preprocess the universal circuit C_U that takes as input a circuit C and an input x and outputs $C(x)$. Example works that follow this approach include [2, 4, 3].

Both these approaches have significant drawbacks. In the first case, the verifier is unable to adaptively change the computation to be verified without re-running the preprocessing phase (which is typically much more costly than evaluating the function). The second approach imposes large concrete overheads, since C_U requires many more gates than C. For implemented systems, the only exception to the above is the recent work of Ben-Sasson et al. [1] that does not utilize a preprocessing phase. However, the concrete cost of their techniques remains significantly higher then the above preprocessing-based approaches.

An Argument System with Circuit Independent Preprocessing. Recently, Zhang et al [30] introduced vSQL, a system for verifying the execution of arbitrary SQL queries over outsourced databases. Their construction combines the "CMT protocol" of Cormode et al. [12] (which itself extends the work of Goldwasser et al. [18]) for verifying the evaluation of arithmetic circuits with a new verifiable polynomial delegation (VPD) scheme in a way that can accommodate auxiliary input. While not explicitly stated in their paper, it can be shown that their underlying construction is an *argument of knowledge* for NP, which is *succinct* if the circuit is "parallelizable" (i.e., can be expressed as parallel copies of smaller circuits, possibly followed by a small "aggregation" circuit), which is the case for SQL queries. An important feature of the argument system resulting from their work is that it has a preprocessing phase that only depends on an upper bound on the size of the input, but not on the specific circuit to be evaluated. This allows obtaining performance which is similar to (1) above, but in the setting of (2) where the circuit to be verified is not known during the preprocessing phase. Another attractive property of vSQL is its improved performance. In contrast to state-of-the-art arguments based on quadratic-arithmetic programs [2, 26], the number of cryptographic operations used by vSQL is linear in the input size (as opposed to linear in the number of multiplication gates of the circuit). As demonstrated in [30] for the case of verifying SQL queries, this translates in practice to a reduction in prover time by 5–121×.

Zero-Knowledge Verifiable Computation. The scheme of [30] lacks one crucial property: it is not *zero-knowledge*. Moreover, while state-of-the-art *succinct non-interactive arguments of knowledge* (SNARKs) from quadratic-arithmetic programs can be made zero-knowledge by simply randomizing proof elements, this approach is not directly compatible with vSQL.

Our Contribution. Here, we show a zero-knowledge version of the argument of knowledge implicitly presented in vSQL [30]. At a high level, our protocol maintains the same structure as vSQL and combines a CMT-like protocol with a VPD scheme. However, in order to obtain a zero-knowledge property we replace both of the underlying components with zero-knowledge variants. For the CMT component, this can be achieved by running the entire protocol inside homomorphic commitments (as first observed in a a more general context by Cramer and Damgård [14]). For the VPD part, we devise a new construction that we call zk-VPD which is used to allow the prover to produce a proof about the correctness of a *commitment* to the correct evaluation of a polynomial (rather than proving correctness of the evaluation itself).

Asymptotically, our protocol has the same performance as that of [30] and has a preprocessing phase that only depends on an upper bound on the size of the input, but not on the specific circuit to be evaluated. In practice, we would expect it to have a slightly larger overhead for both parties (due to the increased number of cryptographic operations).

Other Approaches to Making CMT Zero-knowledge. Chiesa et al. [11] showed how a large class of algebraic protocols (including sum-check and CMT) can be made zero-knowledge using only information theoretic techniques. While this is a very attractive property, it is not clear how to make their approach compatible with a VPD protocol such as the one from [30] and the one we present here. Next, in a concurrent and independent work, Wahby et al. [28] presented an efficient zero-knowledge argument for sufficiently "parallel" circuits that utilizes the CMT protocol and uses the same general approach for making it zero-knowledge as the one used in this work (i.e., running the CMT protocol over homomorphic commitments). Unlike our construction which has a trusted preprocessing phase and relies on non-standard knowledge-of-exponent assumptions, the construction of [28] does not require any preprocessing and its security is based solely on the DDH assumption. However, while our construction

achieves communication size and verification time that are polylogarithmic in the size of the witness w of the NP-relation being verified, the communication size and verification time of [28] scale with $O(\sqrt{|w|})$ which might be prohibitive for some applications.

2 Preliminaries

2.1 Cryptographic Assumptions

We use the following hardness assumption over elliptic curve groups with pairings, originally introduced in [8, 20]

Assumption 1. *(q-Strong Diffie-Hellman) For any probabilistic polynomial time adversary $\mathcal{A}$, the following probability is negligible:*

$$\Pr\left[\begin{array}{c} \mathsf{bp} \leftarrow \mathsf{BilGen}(1^\lambda); \\ s \xleftarrow{R} \mathbb{Z}_p^*; \\ \sigma = (\mathsf{bp}, g^s, \ldots, g^{s^q}) \end{array} : (x, e(g,g)^{\frac{1}{s+x}}) \leftarrow \mathcal{A}(1^\lambda, \sigma)\right].$$

We also use a "knowledge-type" assumption, which is a slightly modified version of an assumption originally introduced in [30]. The latter, in turn, is a generalization of Groth's q-PKE assumption [21] for the case of multivariate polynomials. In the following, $\mathcal{W}_{\ell,d}$ denotes the set of all multisets of $\{1,\ldots,\ell\}$ where the cardinality of each element is at most d.

Assumption 2 ((d,ℓ)-Extended Power Knowledge of Exponent)**.** *For any* PPT *adversary $\mathcal{A}$ there is a polynomial-time algorithm $\mathcal{E}$ (running on the same random tape) such that for all benign auxiliary inputs $z \in \{0,1\}^{poly(\lambda)}$ the following probability is negligible:*

$$\Pr\left[\begin{array}{c} \mathsf{bp} \leftarrow \mathsf{BilGen}(1^\lambda); \\ s_1,\ldots,s_\ell,s_{\ell+1},\alpha \xleftarrow{R} \mathbb{Z}_p^*, s_0 = 1; \\ \sigma_1 = (\{g^{\prod_{i\in W} s_i}\}_{W\in\mathcal{W}_{\ell,d}}, g^{s_{\ell+1}}); \\ \sigma_2 = (\{g^{\alpha\cdot\prod_{i\in W} s_i}\}_{W\in\mathcal{W}_{\ell,d}}, g^{\alpha s_{\ell+1}}); \\ \sigma = (\mathsf{bp},\sigma_1,\sigma_2,g^\alpha); \\ \mathbb{G}\times\mathbb{G} \ni (h,\tilde{h}) \leftarrow \mathcal{A}(1^\lambda,\sigma,z); \\ (a_0,\ldots,a_{|\mathcal{W}_{\ell,d}|},b) \leftarrow \mathcal{E}(1^\lambda,\sigma,z) \end{array} : \begin{array}{c} e(h,g^\alpha) = e(\tilde{h},g) \\ \wedge \\ \prod_{W\in\mathcal{W}_{\ell,d}} g^{a_W \prod_{i\in W} s_i} g^{b s_{\ell+1}} \neq h \end{array}\right].$$

The results of [10, 7] show the impossibility of knowledge assumptions with respect to arbitrary auxiliary inputs. In the above definition we use the notion of a benign auxiliary input (or, alternatively, a benign state generator), similar to [13, 22, 15], to refer to auxiliary inputs that make extraction possible, avoiding these negative results. Concretely, our proofs hold assuming the auxiliary input of the extractor comes from a benign distribution (in practice, the auxiliary inputs in our construction will consist of hiding commitments).

2.2 Interactive Proofs and Argument Systems

Interactive proofs. An interactive proof [19] is a protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ which convinces $\mathcal{V}$ of the validity of a statement $f(x) = 1$ where f and x are common inputs known to both parties.

Definition 1. *A pair of algorithms $(\mathcal{P},\mathcal{V})$ is an interactive proof system for a function f with soundness ϵ if the following properties hold:*

- ***Completeness.*** *For any x such that $f(x) = 1$ it holds that $\Pr[\langle\mathcal{P},\mathcal{V}\rangle(x) = 1] = 1$.*

- ***Soundness.*** *For any x such that $f(x) \neq 1$ and for any prover $\mathcal{P}^*$ it holds that* $\Pr[\langle \mathcal{P}^*, \mathcal{V} \rangle(x) = 1] \leq \epsilon$.

Argument Systems. Let R be an NP relation. An argument system for R is a protocol between computationally bounded prover $\mathcal{P}$ and a verifier $\mathcal{V}$ at the end of which $\mathcal{V}$ is convinced in the validity of a statement made by $\mathcal{P}$ of the from "there exists w such that $(x; w) \in R$" for some input x. In the sequel we focus on *arguments of knowledge* which have the stronger property that if the prover manages to convince the verifier of the statement's validity, then the prover must know w. We use the definition of [17] which includes a parameter-generation phase executed by a trusted party, the preprocessor. Formally, consider Definition 2 below.

Definition 2. *Let R be an NP relation and let λ be a security parameter. A tuple of algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ is a zero knowledge* argument *for R if the following holds.*

- ***Completeness.*** *For every $(\mathsf{pk}, \mathsf{vk})$ outout by $\mathcal{G}(1^\lambda)$ and all $(x; w) \in R$ we have*

$$\langle \mathcal{P}(\mathsf{pk}, w), \mathcal{V}(\mathsf{vk}) \rangle(x) = \mathsf{accept}.$$

- ***Knowledge soundness.*** *For any probabilistic polynomial time prover $\mathcal{P}^*$ there exists a probabilistic polynomial extractor $\mathcal{E}$ which runs on the same randomness as $\mathcal{P}^*$ such that for any x it holds that* $\Pr[\langle \mathcal{P}^*(\mathsf{pk}), \mathcal{V}(\mathsf{vk}) \rangle(x) = \mathsf{accept} \wedge (x, w) \notin L | (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{G}(1^\lambda), w \leftarrow \mathcal{E}(\mathsf{pk}, x)] \leq \mathsf{neg}(\lambda)$.

- ***Zero knowledge.*** *There exists a probabilistic polynomial simulator $\mathcal{S}$ such that for any probabilistic polynomial time adversary $\mathcal{A}$, auxiliary input $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$ the following holds*

$$\Pr\left[(x; w) \in R; \langle \mathcal{P}(\mathsf{pk}, w), \mathcal{A}(\sigma) \rangle(x) = \mathsf{accept} : (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{G}(1^\lambda); (x, w, \sigma) \leftarrow \mathcal{A}(z, \mathsf{pk}, \mathsf{vk})\right] \approx$$
$$\Pr\left[(x; w) \in R; \langle \mathcal{S}(\mathsf{trap}, z, \mathsf{pk}), \mathcal{A}(\sigma) \rangle(x) = \mathsf{accept} : (\mathsf{pk}, \mathsf{vk}, \mathsf{trap}) \leftarrow \mathcal{S}(1^\lambda); (x, w, \sigma) \leftarrow \mathcal{A}(z, \mathsf{pk}, \mathsf{vk})\right]$$

where $\approx$ denotes computational indistinguishability (and the definition can be extended in a straightforward manner for statistical and perfect zero-knowledge).

We call $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ a succinct *argument system if the running time of $\mathcal{V}$ is $poly(\lambda, |x|, \log |w|)$.*

2.3 Building Blocks

Linearly homomorphic commitment scheme. We assume the existence of a commitment scheme $\mathcal{C}omm = (\mathcal{S}etup, \mathcal{C}om, \mathcal{O}pen)$ that has $\mathbb{Z}_q$ (for prime q) as its message space. This could be instantiated by the Pedersen commitment scheme [27], for example. We assume:

- $\mathcal{S}etup(1^\lambda)$ outputs public commitment parameters cp.
- $\mathcal{C}om(\mathsf{cp}, m, r)$ on input a message $m \in \mathbb{Z}_q$ and randomness r outputs a commitment com.
- $\mathcal{O}pen(\mathsf{cp}, \mathsf{com}, m, r)$ accepts iff $\mathcal{C}om(\mathsf{cp}, m, r) = \mathsf{com}$.

We also require that there exists an efficient algorithm $\mathcal{E}val(\mathsf{cp}, \mathsf{com}_1, \ldots, \mathsf{com}_n, x_1, \ldots, x_n)$ that on input n valid commitments (for some randomness values r_i) for $m_1, \ldots, m_n$ and coefficients $x, \ldots, x_n \in \mathbb{F}$, outputs new commitment com' such that $\mathcal{O}pen(\mathsf{cp}, \mathsf{com}', \sum_1^n x_i m_i, r')$ accepts, where r' is computed as a function of $(r_1, \ldots, r_n, x_1, \ldots, x_n)$. For Pedersen commitments, this can be easily achieved by having $\mathcal{E}val(\mathsf{cp}, \mathsf{com}_1, \ldots, \mathsf{com}_n, x_1, \ldots, x_n)$ outputs $\mathsf{com}' = \prod_1^n \mathsf{com}_i^{x_i}$ and $r' = \sum_1^n x_i r_i$.

Zero-knowledge argument for commitment-preimage equality. We assume the existence of a zero-knowledge argument $\mathcal{ZK}_{eq}$ for proving that two commitments produced with $\mathcal{C}omm$ have the same pre-image. Somewhat informally, we write $\mathcal{ZK}_{eq}(m, r_1, r_2; \mathsf{com}_1, \mathsf{com}_2) \rightarrow \mathsf{accept/reject}$ to denote the

interaction between a prover that holds $\mathsf{cp}, m, r_1, r_2, \mathsf{com}_1, \mathsf{com}_2$ such that $\mathcal{O}pen(\mathsf{cp}, \mathsf{com}_i, m, r_i)$ accepts for $i = 1, 2$, and a verifier that holds $\mathsf{cp}, \mathsf{com}_1, \mathsf{com}_2$ will eventually accept if he believes they have the same preimage and he will reject otherwise. For completeness, we require that the verifier accepts with probability 1 for a valid statement. For soundness, we require that for any probabilistic polynomial-time (cheating) prover algorithm, the verifier will accept a false statement with probability negligible in λ. Zero-knowledge dictates that the verifier learns nothing about m_1, m_2. For the Pedersen commitment scheme, such a protocol can be instantiated by first using a sigma-protocol (e.g., the one from [9]) and then using standard techniques to make it full zero-knowledge (e.g., [16]).

Zero-knowledge argument for product of preimages. We assume the existence of a zero-knowledge argument $\mathcal{ZK}_{prod}$ for proving that for three commitments $\mathsf{com}_1, \mathsf{com}_2, \mathsf{com}_3$ produced with $\mathcal{C}omm$ it holds that the preimage of the last is the product (in $\mathbb{F}$) of the preimages of the first two. We write $\mathcal{ZK}_{eq}(m_1, m_2, r_1, r_2, r_3; \mathsf{com}_1, \mathsf{com}_2, \mathsf{com}_3) \rightarrow \mathsf{accept/reject}$ to denote the interaction between a prover that holds $\mathsf{cp}, m_1, m_2, r_1, r_2, r_3, \mathsf{com}_1, \mathsf{com}_2, \mathsf{com}_3$ such that $\mathcal{O}pen(\mathsf{cp}, \mathsf{com}_i, m_i, r_i)$ accepts for $i = 1, 2$, and $\mathcal{O}pen(\mathsf{cp}, \mathsf{com}_3, m_1 \cdot m_2, r_3)$ accepts, and a verifier that holds $\mathsf{cp}, \mathsf{com}_1, \mathsf{com}_2, \mathsf{com}_3$. For completeness, we require that the verifier accepts with probability 1 for a valid statement. For soundness, we require that for any probabilistic polynomial-time (cheating) prover algorithm, the verifier will accept a false statement with probability negligible in λ. Zero-knowledge dictates that the verifier learns nothing about m_1, m_2. For the Pedersen commitment scheme, this can again be instantiated via a standard combination of [9, 16].

Exractability. Finally, we want the commitment scheme to be *extractable* in the manner described in [6] for the case of collision-resistant functions, i.e., it should not be possible to output a valid commitment without knowing a corresponding pre-image. Somewhat informally, this is captured by the existence of an adversary-specific extractor that (given access to the adversary's code, random tape and auxiliary input) can output a pre-image for any commitment value the adversary produces with all but negligible probability. For the Pedersen commitment scheme this can be achieved, under Assumption 2, via the following modifications. (1) Parameters cp also include value g^β for $\beta \in \mathbb{F}$ chosen uniformly at random. (2) Commitments consist of a pair of values from $\mathsf{com}, \mathsf{com}' \in \mathbb{G}$ such that $\mathsf{com}' = \mathsf{com}^\beta$. (3) Upon receiving such a commitment $\mathsf{com}, \mathsf{com}'$, the receiving party must check the relation $e(\mathsf{com}, g^\beta) = e(\mathsf{com}, g)$ and abort if the check fails. To ease notation, in the following when describing a commitment value we will only refer to com and we will omit the above validity check from the description of our protocols.

2.4 Circuit and Polynomial Notation

Let C be an arithmetic circuit. We denote the number of gates in the i-th layer of C by S_i and we set $s_i = \lceil \log S_i \rceil$ (that is, s_i bits are sufficient in order to uniquely identify each gate is the i-th layer). The evaluation of C on an input x assigns (in the natural way) a value from $\mathbb{F}$ to each gate in C. For each layer i in C, we define the function $V_i : \{0,1\}^{s_i} \rightarrow \mathbb{F}$ that takes as input a gate $g \in \{0,1\}^{s_i}$ and outputs its value. Note that the values returned by V_d correspond to the values of the input layer of C, i.e., x. Finally, for each layer i we define functions add_i, mult_i to which we refer as C's *wiring predicates*. The function $\mathsf{add}_i : \{0,1\}^{s_{i-1}+2s_i} \rightarrow \{0,1\}$ takes as input a gate g_1 from layer $i-1$ and two gates g_2, g_3 from layer i and outputs 1 if g_1 is an addition gate whose inputs are connected to g_2 and g_3. The function mult_i is defined similarly for multiplication gates. Finally, we notice that the value of a gate g at layer $i < d$ can be computed as a function of the values of gates at layer $i+1$ as

$$V_i(g) = \sum_{u,v \in \{0,1\}^{s_{i+1}}} (\mathsf{add}_{i+1}(g,u,v) \cdot (V_{i+1}(u) + V_{i+1}(v)) + \mathsf{mult}_{i+1}(g,u,v) \cdot (V_{i+1}(u) \cdot V_{i+1}(v))).$$

In the following, we define the *variable-degree* of a multivariate polynomial f to be the maximum degree of f in any of its variables, and use $\mathcal{W}_{\ell,d}$ to denote the collection of all multisets of $\{1, \ldots, \ell\}$ for which the multiplicity of any element is at most d. The following useful lemma is due to Papamanthou et al. [25].

Lemma 1 ([25]). *Let $f : \mathbb{F}^\ell \to \mathbb{F}$ be a polynomial of variable degree d. For all $t \in \mathbb{F}^\ell$ there exist efficiently computable polynomials $q_1, \ldots, q_\ell$ such that: $f(x) - f(t) = \sum_{i=1}^{\ell}(x_i - t_i)q_i(x)$ where t_i is the ith element of t.*

3 Zero-Knowledge Polynomial Delegation Commitment Scheme

In this section we present our zero knowledge polynomial delegation scheme. At a high level, the main idea is to modify the polynomial delegation scheme of [30] to output a commitment to the evaluation instead of the evaluation itself. That is, instead of having $\mathsf{Evaluate}$ output the value y of the polynomial f when evaluated on the input x together with a suitable proof π, the zero-knowledge polynomial delegation commitment scheme outputs a statistically hiding and computationally binding commitment com_y to the value of y (in addition to the proof π). This hides the value of y but still supports verifying that com_y is indeed a commitment to $f(x)$.

Figure 3 contains our definition of a zero-knowledge polynomial delegation scheme. Next, consider the following theorem.

Theorem 1. *Under Assumptions 1 and 2, Construction 1 is a zero-knowledge extractable verifiable polynomial-delegation protocol. Moreover, for a variable-degree-d ℓ-variate polynomial $f \in \mathcal{F}$ containing m monomials, algorithm KeyGen runs in time $O(\binom{\ell(d+1)}{\ell d})$, $\mathsf{CommitPoly}$ in time $O(m)$, $\mathsf{CommitValue}$ in time $O(\ell dm)$, Ver in time $O(\ell)$ and $\mathsf{CheckCom}$ in time $O(1)$. If $d = 1$, $\mathsf{CommitValue}$ can be made to run in time $O(2^\ell)$. The commitment produced by $\mathsf{CommitPoly}$ consists of $O(1)$ group elements, and the proof produced by $\mathsf{CommitPoly}$ consists of $O(\ell)$ elements of $\mathbb{G}$.*

Proof. Completeness follows by close inspection of the algorithms. Next, we prove the rest of the properties of Definition 3.

Polynomial Extractability. Let $\mathcal{A}$ be a PPT adversary that on input $(1^\lambda, \mathsf{pp})$, where $(\mathsf{pp}, \mathsf{vp})$ is the output of $\mathsf{KeyGen}(1^\lambda, \ell, d)$, outputs commitment com_f^* such that $\mathsf{CheckCom}(\mathsf{com}_f^*, \mathsf{vp})$ accepts. This implies that $e(\mathsf{com}_{f,1}, g^\alpha) = e(\mathsf{com}_{f,2}, g)$. By Assumption 2, there exists PPT extractor $\mathcal{E}$ for $\mathcal{A}$ such that upon the same input as $\mathcal{A}$, and with access to same random tape, outputs $a_0, \ldots, a_{|\mathcal{W}_{\ell,d}|}, b \in \mathbb{F}$ such that $\prod_{W \in \mathcal{W}_{\ell,d}} g^{a_W \prod_{i \in W} s_i} g^{b s_{\ell+1}} = \mathsf{com}_{f,1}$, except with negligible probability. Note that, the coefficients $(a_0, \ldots, a_{|\mathcal{W}_{\ell,d}|}, b)$ can always be encoded as an $(\ell+1)$-variate polynomial that consist of the sum of two polynomials: an ℓ-variate one with degree-variable d that is defined over variables $x_1, \ldots, x_\ell$ and has values a_i as its monomial coefficients, and the univariate, degree-1 polynomial $bx_{\ell+1}$.

Binding. Next, we prove the binding property. Let $\mathcal{A}$ be a PPT adversary that wins the binding game with non-negligible probability. For $i = 1, \ldots, \ell+1$ we define adversary $\mathcal{A}_i$ that receives the same input as $\mathcal{A}$ and executes the same code, but outputs only $\mathsf{com}_i \in \pi^*$ (where π^* is the proof output by $\mathcal{A}$). Moreover, since $\mathcal{A}$ is PPT, all these adversaries are also PPT. Thus, for $i = 1, \ldots, \ell+1$, from Assumption 2 there exists PPT $\mathcal{E}_i$ (running on the same random tape as $\mathcal{A}_i$) which on input $(1^\lambda, \mathsf{pp})$ outputs $a_{0,i}, \ldots, a_{|\mathcal{W}_{\ell,d}|,i}, b_i \in \mathbb{F}$ such that the following holds: If $e(\mathsf{com}_{i,1}, g^\alpha) = e(\mathsf{com}_{i,2}, g)$ then $\prod_{W \in \mathcal{W}_{\ell,d}} g^{a_{W,i} \prod_{j \in W} s_j} \cdot g^{b_i s_{\ell+1}} \neq \mathsf{com}_{i,1}$, except with negligible probability. By the same reasoning as above, the coefficients $(a_{0,i}, \ldots, a_{|\mathcal{W}_{\ell,d}|,i} b_i)$ for each $i = 1, \ldots, \ell$ can always be encoded as an $(\ell+1)$-variate q_i' that can be expressed as the sum of an ℓ-variate polynomial with variable-degree d that is defined over variables $x_1, \ldots, x_\ell$ and a univariate degree-1 polynomial defined over $x_{\ell+1}$.

We now proceed to build an adversary $\mathcal{B}$ that breaks Assumption 1 for parameter $(\ell+1) \cdot d$. Upon input $(1^\lambda, \mathsf{bp}, g^s, g^{s^2}, \ldots, g^{s^{(\ell+1)\cdot d}})$, $\mathcal{B}$ proceeds as follows:

Parameter Generation. $\mathcal{B}$ implicitly sets $s_1 = s$ and for $i = 2, \ldots, \ell+1$ he chooses $\rho_i \in \mathbb{F}$ uniformly at random and sets (also implicitly) $s_i = s \cdot \rho_i$. Then he chooses uniformly at random values $\alpha, \beta \in \mathbb{F}$. Next $\mathcal{B}$ needs to generate the terms in $\mathbb{P} = \{g^{\prod_{i \in W} s_i}, g^{\alpha \cdot \prod_{i \in W} s_i}\}_{W \in \mathcal{W}_{\ell,d}}$. Since the exponent of each term is a product of at most $\ell \cdot d$ factors where each factor is one of the values $s_i = s \cdot r_i$ (for $i = 1, \ldots, \ell$), it can

Definition 3. *Let $\mathbb{F}$ be a finite field, $\mathcal{F}$ be a family of ℓ-variate polynomials over $\mathbb{F}$, and d be a variable-degree parameter.* $(\mathsf{KeyGen}, \mathsf{Commit}, \mathsf{Evaluate}, \mathsf{CheckCom}, \mathsf{Ver})$ *constitute a* zero-knowledge verifiable polynomial-delegation protocol *for $\mathcal{F}$ if:*

- ***Perfect Completeness.*** *For any polynomial $f \in \mathcal{F}$ and value t,*

$$\Pr_{r_f, r_y}\left[\begin{array}{c}(\mathsf{pp},\mathsf{vp}) \leftarrow \mathsf{KeyGen}(1^\lambda, \ell, d) \\ \mathsf{com}_f \leftarrow \mathsf{CommitPoly}(f, r_f, \mathsf{pp}) \\ (\mathsf{com}_y, \pi) \leftarrow \mathsf{CommitValue}(f, t, f(t), r_f, r_y, \mathsf{pp})\end{array} : \begin{array}{c}\mathsf{CheckCom}(\mathsf{com}_f, \mathsf{vp}) = \mathtt{accept} \wedge \\ \mathsf{Ver}(\mathsf{com}_f, t, \mathsf{com}_y, \pi, \mathsf{vp}) = \mathtt{accept}\end{array}\right] = 1.$$

- ***Binding.*** *For any* PPT *adversary $\mathcal{A}$ and benign auxilary inputs z_1, z_2 the following probability is negligible:*

$$\Pr\left[\begin{array}{c}(\mathsf{pp},\mathsf{vp}) \leftarrow \mathsf{KeyGen}(1^\lambda, \ell, d) \\ (\pi^*, \mathsf{com}_f^*, \mathsf{com}_y^*, \mathsf{state}) \leftarrow \mathcal{A}(1^\lambda, z_1, \mathsf{pp}) \\ (f^*, t^*, y^*, r_f^*, r_y^*) \leftarrow \mathcal{A}(1^\lambda, z_2, \mathsf{state}, \mathsf{pp})\end{array} : \begin{array}{c}\mathsf{CheckCom}(\mathsf{com}_f^*, \mathsf{vp}) = \mathtt{accept} \wedge \\ \mathsf{Ver}(\mathsf{com}_f^*, t^*, \mathsf{com}_y^*, \pi^*, \mathsf{vp}) = \mathtt{accept} \wedge \\ \mathsf{com}_f^* = \mathsf{CommitPoly}(f^*, r_f^*, \mathsf{pp}) \wedge \\ (\mathsf{com}_y^*, \pi) = \mathsf{CommitValue}(f^*, t^*, y^*, r_f^*, r_y^*, \mathsf{pp}) \wedge f^*(t^*) \neq y^*\end{array}\right].$$

- ***Zero Knowledge.*** *For security parameter λ, polynomial f, adversary $\mathcal{A}$, and simulator* Sim *consider the two experiments $\mathsf{Real}_{\mathcal{A},f}(1^\lambda), \mathsf{Ideal}_{\mathcal{A}}(1^\lambda)$, defined as follows.*

$\underline{\mathsf{Real}_{\mathcal{A},f}(1^\lambda)}$*:*

1. $(\mathsf{pp},\mathsf{vp}) \leftarrow \mathsf{KeyGen}(1^\lambda, \ell, d)$
2. *Generate r_f uniformly at random*
3. $\mathsf{com}_f \leftarrow \mathsf{CommitPoly}(f, r_f, \mathsf{pp})$
4. $k \leftarrow \mathcal{A}(1^\lambda, \mathsf{com}_f, \mathsf{vp})$
5. *For $i = 1, \ldots, k$ repeat:*

(a) *Generate r_i uniformly at random*

(b) $t_i \leftarrow \mathcal{A}(1^\lambda, \mathsf{com}_f, \mathsf{com}_{y_1} \ldots, \mathsf{com}_{y_{i-1}}, \pi_1, \ldots, \pi_{i-1}, \mathsf{vp})$

(c) $(\mathsf{com}_{y_i}, \pi_i) \leftarrow \mathsf{CommitValue}(f, t_i, f(t_i), r_f, r_i, \mathsf{pp})$

6. $b \leftarrow \mathcal{A}(1^\lambda, \mathsf{com}_f, (\mathsf{com}_{y_1} \ldots, \mathsf{com}_{y_k}, \pi_1, \ldots, \pi_k), \mathsf{vp})$
7. *Output b*

$\underline{\mathsf{Ideal}_{\mathcal{A},\mathsf{Sim}}(1^\lambda)}$*:*

1. $(\mathsf{com}_f, \mathsf{pp}, \mathsf{vp}, \sigma) \leftarrow \mathsf{Sim}(1^\lambda, \ell, d)$
2. $k \leftarrow \mathcal{A}(1^\lambda, \mathsf{com}_f, \mathsf{vp})$
3. *For $i = 1, \ldots, k$ repeat:*

(a) $t_i \leftarrow \mathcal{A}(1^\lambda, \mathsf{com}_f, \mathsf{com}_{y_1} \ldots, \mathsf{com}_{y_{i-1}}, \pi_1, \ldots, \pi_{i-1}, \mathsf{vp})$

(b) $(\mathsf{com}_{y_i}, \pi_i, \sigma) \leftarrow \mathsf{Sim}(t_i, \sigma, \mathsf{pp})$

4. $b \leftarrow \mathcal{A}(1^\lambda, \mathsf{com}_f, (\mathsf{com}_{y_1} \ldots, \mathsf{com}_{y_k}, \pi_1, \ldots, \pi_k), \mathsf{vp})$
5. *Output b*

We require that for any PPT *adversary $\mathcal{A}$ and all $f \in \mathbb{F}$, there exists a simulator* Sim *such that the following is negligible*

$$\left|\Pr\left[\mathsf{Real}_{\mathcal{A},f}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Ideal}_{\mathcal{A},\mathsf{Sim}}(1^\lambda) = 1\right]\right|.$$

Finally, we say that $(\mathsf{KeyGen}, \mathsf{Commit}, \mathsf{Evaluate}, \mathsf{CheckCom}, \mathsf{Ver})$ *are an* extractable *zero-knowledge verifiable polynomial-delegation protocol for $\mathcal{F}$ if* $(\mathsf{KeyGen}, \mathsf{Commit}, \mathsf{Evaluate}, \mathsf{CheckCom}, \mathsf{Ver})$ *satisfy the following extraction requirements instead of the above defined soundness requirement.*

- ***Polynomial Extractability.*** *For any* PPT *adversary $\mathcal{A}$ there exists a polynomial-time algorithm $\mathcal{E}$ with access to $\mathcal{A}$'s random tape such that for all benign auxiliary inputs $z \in \{0,1\}^{poly(\lambda)}$ the following probability is negligible:*

$$\Pr\left[\begin{array}{c}(\mathsf{pp},\mathsf{vp}) \leftarrow \mathsf{KeyGen}(1^\lambda, \ell, d); \\ \mathsf{com}_f^* \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}, z); \\ (f, r_f) \leftarrow \mathcal{E}(1^\lambda, \mathsf{pp}, z)\end{array} : \begin{array}{c}\mathsf{CheckCom}(\mathsf{com}^*, \mathsf{vp}) = \mathtt{accept} \wedge \\ \mathsf{com}_f^* \neq \mathsf{CommitPoly}(f, r_f, \mathsf{pp})\end{array}\right].$$

- ***Evaluation Extractability.*** *For any* PPT *adversary $\mathcal{A}$ there exists a polynomial-time algorithm $\mathcal{E}$ with access to $\mathcal{A}$'s random tape such that for all benign auxiliary inputs $z \in \{0,1\}^{poly(\lambda)}$ the following probability is negligible:*

$$\Pr\left[\begin{array}{c}(\mathsf{pp},\mathsf{vp}) \leftarrow \mathsf{KeyGen}(1^\lambda, \ell, d) \\ (t^*, \pi^*, \mathsf{com}_f^*, \mathsf{com}_y^*) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}, z) \\ (f, r_f, y, r_y) \leftarrow \mathcal{E}(1^\lambda, \mathsf{pp}, z)\end{array} : \begin{array}{c}\mathsf{CheckCom}(\mathsf{com}_f^*, \mathsf{vp}) = \mathtt{accept} \wedge \\ \mathsf{Ver}(\mathsf{com}_f^*, t^*, \mathsf{com}_y^*, \pi^*, \mathsf{vp}) = \mathtt{accept} \wedge \\ (f(t^*) \neq y \ \vee \ \mathsf{com}_f^* \neq \mathsf{CommitPoly}(f, r_f, \mathsf{pp}) \vee \\ (\pi^*, \mathsf{com}_y^*) \neq \mathsf{CommitValue}(f, t^*, y, r_f, r_y, \mathsf{pp}))\end{array}\right].$$

Construction 1 (Zero-knowledge Verifiable Polynomial-Delegation Protocol). *Let $\mathbb{F}$ be a prime-order finite field, ℓ be a variable parameter, and d be a variable-degree parameter such that $O(\binom{\ell(d+1)}{\ell d})$ is polynomial in λ. Consider the following protocol for the family $\mathcal{F}$ containing ℓ-variate polynomials of variable-degree d over $\mathbb{F}$.*

1. **KeyGen($1^\lambda, \ell, d$):** *Select $\alpha, \beta, s_1, \ldots, s_\ell, s_{\ell+1} \in \mathbb{F}$ uniformly at random, run $\mathsf{bp} \leftarrow \mathsf{BilGen}(1^\lambda)$ and compute $\mathbb{P} = \{g^{\prod_{i \in W} s_i}, g^{\alpha \cdot \prod_{i \in W} s_i}\}_{W \in \mathcal{W}_{\ell,d}}$. The public parameters are set to be $\mathsf{pp} = (\mathsf{bp}, \mathbb{P}, g^\alpha, g^\beta, g^{s_{\ell+1}}, g^{\alpha s_{\ell+1}}, g^{\beta s_{\ell+1}})$ and the verifier parameters are set to be $\mathsf{vp} = (\mathsf{bp}, g^{s_1}, \cdots, g^{s_\ell}, g^{s_{\ell+1}}, g^\alpha, g^\beta)$.*

2. **CommitPoly(f, r_f, pp):** *If $f \notin \mathcal{F}$ output null. Else, compute $c_1 = g^{f(s_i,\ldots,s_\ell)+r_f s_{\ell+1}}$ and $c_2 = g^{\alpha \cdot (f(s_i,\ldots,s_\ell)+r_f s_{\ell+1})}$, and output the commitment $\mathsf{com}_f = (c_1, c_2)$.*

3. **CheckCom($\mathsf{com}_f, \mathsf{vp}$):** *On input a commitment $\mathsf{com}_f = (\mathsf{com}_{f,1}, \mathsf{com}_{f,2})$, check whether it is well-formed, i.e., output accept if $e(\mathsf{com}_{f,1}, g^\alpha) = e(\mathsf{com}_{f,2}, g)$ and reject otherwise.*

4. **CommitValue($f, t, y, r_f, r_y, \mathsf{pp}$):** *Choose $r_1, \ldots, r_\ell \in \mathbb{F}$ uniformly at random. Next, using Lemma 1 compute polynomials q_i such that*

$$f(x_1, \ldots, x_\ell) + r_f x_{\ell+1} - (y + r_y x_{\ell+1}) = \sum_{i=1}^{\ell} (x_i - t_i) \cdot (q_i(x_i, \ldots, x_\ell) + r_i x_{\ell+1}) + x_{\ell+1}(r_f - r_y - \sum_{i=1}^{\ell} r_i (x_i - t_i)).$$

   Set $\mathsf{com}_y \leftarrow (g^{y + r_y s_{\ell+1}}, g^{\beta y + \beta r_y s_{\ell+1}})$. For $i = 1, \ldots, \ell$ compute $\mathsf{com}_i \leftarrow \mathsf{CommitPoly}(q_i, r_i, \mathsf{pp})$. Compute $\mathsf{com}_{\ell+1} \leftarrow (g^{r_f - r_y - \sum_{i=1}^{\ell} r_i (s_i - t_i)}, g^{\alpha(r_f - r_y - \sum_{i=1}^{\ell} r_i (s_i - t_i))})$. Output com_y and the proof $\pi := (\mathsf{com}_1, \ldots, \mathsf{com}_{\ell+1})$.

5. **Ver($\mathsf{com}_f, t, \mathsf{com}_y, \pi, \mathsf{vp}$):** *Parse the proof π as $(\mathsf{com}_1, \ldots, \mathsf{com}_{\ell+1})$. For $i = 1, \ldots, \ell + 1$ run $\mathsf{CheckCom}(\mathsf{com}_i, \mathsf{pp})$. If any of them outputs reject, output reject. Otherwise, parse com_f as $(\mathsf{com}_{f,1}, \mathsf{com}_{f,2})$ and com_y as $(\mathsf{com}_{y,1}, \mathsf{com}_{y,2})$ and for $i = 1, \ldots, \ell + 1$ parse com_i as $(\mathsf{com}_{i,1}, \mathsf{com}_{i,2})$. If $e(\mathsf{com}_{y,1}, g^\beta) \stackrel{?}{=} e(\mathsf{com}_{y,2}, g)$ and $e(\mathsf{com}_{f,1}/\mathsf{com}_{y,1}, g) \stackrel{?}{=} e(g^{s_{\ell+1}}, \mathsf{com}_{\ell+1,1}) \prod_{i=1}^{\ell} e(g^{s_i - t_i}, \mathsf{com}_i)$ output accept, otherwise output reject.*

be written as a polynomial in s with degree at most $\ell \cdot d$. Therefore, $\mathcal{B}$ can compute these terms from the values $g, g^s, g^{s^2}, \ldots, g^{s^{\ell \cdot d}}$ and α. Then, he computes $g^{s_{\ell+1}}, g^{\alpha s_{\ell+1}}, g^{\beta s_{\ell+1}}$. Finally, $\mathcal{B}$ runs $\mathcal{A}$ on input $(1^\lambda, \mathsf{pp})$, where $\mathsf{pp} = (\mathsf{bp}, \mathbb{P}, g^\alpha, g^\beta, g^{s_{\ell+1}}, g^{\alpha s_{\ell+1}}, g^{\beta s_{\ell+1}})$.

Query Evaluation. Upon eventually receiving $(f^*, t^*, y^*, \pi^*, \mathsf{com}_f^*, \mathsf{com}_y^*, r_f^*, r_y^*)$ from $\mathcal{A}$, $\mathcal{B}$ first checks whether $\mathsf{CommitPoly}(f^*, r_f^*, \mathsf{pp}) = \mathsf{com}_f^*$ and $\mathsf{CommitValue}(f^*, t^*, r_y^*, \mathsf{pp}) = \mathsf{com}_f^*$ and aborts if any of the checks fails. Then, he runs $\mathsf{Ver}(\mathsf{com}_f^*, t^*, \mathsf{com}_y^*, \pi^*, \mathsf{vp})$ where $\mathsf{vp} = (\mathsf{bp}, g^{s_1}, \ldots, g^{s^{\ell+1}}, g^\alpha, g^\beta)$. If Ver rejects $\mathcal{B}$ aborts, else he runs extractors $\mathcal{E}_1, \ldots, \mathcal{E}_{\ell+1}$ (defined above) on the same input as $\mathcal{A}$ and receives polynomials $q'_1, \ldots, q'_{\ell+1}$.

If for the output of any of the $\mathcal{E}_i$ it holds that $\prod_{W \in \mathcal{W}_{\ell,d}} g^{a_{W,i} \prod_{j \in W} s_j} g^{b_i s_{\ell+1}} \neq \mathsf{com}_{i,1}$, $\mathcal{B}$ aborts. Let $\delta = y^* - f^*(t^*)$. If $\delta = 0$ (i.e., $y^* = f^*(t^*)$), $\mathcal{B}$ aborts.

Otherwise, let $K(\mathbf{x}) \stackrel{\text{def}}{=} f^*(\mathbf{x}) - \sum_{i=1}^{\ell}(x_i - t_i) q'_i(\mathbf{x}) - x_{\ell+1} q'_{\ell+1}(\mathbf{x}) + (r_f - r_y) x_{\ell+1} - f^*(t^*)$. Note that by setting $s_1 = s, s_2 = \rho_2 \cdot s, \ldots, s_{\ell+1} = \rho_{\ell+1} \cdot s$, we implicitly set variables $x_2, \ldots, x_{\ell+1}$ to $\rho_2 \cdot x_1, \ldots, x_\ell = \rho_{\ell+1} \cdot x_1$. Thus, $K(\mathbf{x})$ can be interpreted as an (efficiently computable) univariate polynomial of degree at most $(\ell + 1) \cdot d$ over variable x_1, which we refer to as $K'(x_1)$.

$\mathcal{B}$ then proceeds as follows. He chooses $\tau \in \mathbb{F}$ uniformly at random. If $g^\tau = g^{-s}$, he aborts. Else, he computes univariate polynomial Q of degree at most $(\ell + 1) \cdot d$ and value $R \in \mathbb{F}$ such that $K'(x_1) = (x_1 + \tau) Q(x_1) + R$. We then distinguish two cases. (1) If $R = \delta$ then $\mathcal{B}$ factorizes the polynomial K' and let $Y \subset \mathbb{F}$ be the set of its roots ($|Y| \leq (\ell + 1) \cdot d$). For each $y \in Y$, $\mathcal{B}$ tests whether $g^y = g^s$. If so, he outputs $(\tau, e(g, g)^{\frac{1}{y+\tau}})$ as a challenge tuple for Assumption 1 and halts. If all these checks fail, he aborts. (2) Else, (if $R \neq \delta$) he outputs $(\tau, e(g, g)^{Q(s_1) \cdot (\delta - R)^{-1}})$ as a challenge tuple for Assumption 1 and halts. Recall that, (as explained above) the expression in the exponent is a $(\ell + 1) \cdot d$ degree polynomial thus

the challenge value is computable in polynomial time from $(1^\lambda, p, \mathbb{G}, \mathbb{G}_T, e, g, g^s, g^{s^2}, \ldots, g^{s^{(\ell+1)\cdot d}})$.

$\mathcal{B}$ is clearly PPT since all of $\mathcal{E}_i$ are PPT and he performs polynomially many operations in $\mathbb{F}, \mathbb{G}, \mathbb{G}_T$. Next, we analyze the success probability of $\mathcal{B}$. Recall that, by assumption $\mathcal{A}$ succeeds in breaking the binding property of the scheme with non-negligible probability ϵ. We observe that, *conditioned on not aborting*, $\mathcal{B}$ perfectly emulates the binding game to $\mathcal{A}$ and moreover $\mathcal{B}$'s output is always a valid tuple for breaking Assumption 1. Let us argue why this is true.

Since verification succeeded, it holds that

$$e(\mathsf{com}_{f,1}/\mathsf{com}_{y,1}, g) = e(g^{s_{\ell+1}}, \mathsf{com}_{\ell+1,1}) \prod_{i=1}^{\ell} e(g^{s_i - t_i}, \mathsf{com}_{i,1})$$

and since extraction succeeded this can be replaced with

$$\begin{aligned}
e(g,g)^{f^*(s_1,\ldots,s_\ell)+r_f x_{\ell+1}-y^*-r_y x_{\ell+1}} &= e(g,g)^{s_{\ell+1}q'_{\ell+1}(s_1,\ldots,s_{\ell+1})} \prod_{i=1}^{\ell} e(g,g)^{(s_i-t_i)q'_i(s_1,\ldots,s_{\ell+1})} \\
e(g,g)^{f^*(s_1,\ldots,s_\ell)-y^*} &= e(g,g)^{s_{\ell+1}q'_{\ell+1}(s_1,\ldots,s_{\ell+1})} \prod_{i=1}^{\ell} e(g,g)^{(s_i-t_i)q'_i(s_1,\ldots,s_{\ell+1})} e(g,g)^{(r_y-r_f)x_{\ell+1}} \\
e(g,g)^{f^*(s_1,\ldots,s_\ell)-y^*} &= e(g,g)^{s_{\ell+1}q'_{\ell+1}(s_1,\ldots,s_{\ell+1})+(r_y-r_f)x_{\ell+1}+\sum_{i=1}^{\ell}(s_i-t_i)q'_i(s_1,\ldots,s_{\ell+1})} \\
e(g,g)^{-y^*} &= e(g,g)^{s_{\ell+1}q'_{\ell+1}(s_1,\ldots,s_{\ell+1})+(r_y-r_f)x_{\ell+1}+\sum_{i=1}^{\ell}(s_i-t_i)q'_i(s_1,\ldots,s_{\ell+1})-f^*(s_1,\ldots,s_\ell)} \\
e(g,g)^{-\delta-f^*(t^*)} &= e(g,g)^{s_{\ell+1}q'_{\ell+1}(s_1,\ldots,s_{\ell+1})+(r_y-r_f)x_{\ell+1}+\sum_{i=1}^{\ell}(s_i-t_i)q'_i(s_1,\ldots,s_{\ell+1})-f^*(s_1,\ldots,s_\ell)} \\
e(g,g)^{-\delta} &= e(g,g)^{s_{\ell+1}q'_{\ell+1}(s_1,\ldots,s_{\ell+1})+(r_y-r_f)x_{\ell+1}+\sum_{i=1}^{\ell}(s_i-t_i)q'_i(s_1,\ldots,s_{\ell+1})-f^*(s_1,\ldots,s_\ell)+f^*(t^*)} \\
e(g,g)^{\delta} &= e(g,g)^{K(s_1,\ldots,s_{\ell+1})} \\
e(g,g)^{\delta} &= e(g,g)^{K'(s_1)} = e(g,g)^{(x_1+tau)Q(s_1)+R}
\end{aligned}$$

In order for the last substitution to be possible, it must the case that $K'(x_1)$, and correspondingly $K'(\mathbf{x})$ is non-constant polynomial (i.e., with degree > 0). Recall, that for polynomials defined over finite fields division is always possible assuming that the divident's degree is at least as large as that of the divisor's. Moreover, the degree of the quotient is at most that of the divident's and that of the remainder is strictly smaller than that of the divisor (i.e., R is a constant in this case).

Let us assume that $K'(\mathbf{x})$ is a constant polynomial. Since, $e(g,g)^\delta = e(g,g)^{K(s_1,\ldots,s_{\ell+1})}$ and $e(g,g)$ is a generator or $\mathbb{G}_T$, it must be that $K'(\mathbf{x}) \stackrel{\text{def}}{=} \delta$ therefore we can write

$$\begin{aligned}
f^*(x_1,\ldots,x_\ell) - \delta - f^*(x_1,\ldots,x_\ell) &= x_{\ell+1}q'_{\ell+1}(x_1,\ldots,x_{\ell+1}) + (r_y - r_f)x_{\ell+1} + \sum_{i=1}^{\ell}(x_i - t_i)q'_i(x_1,\ldots,x_{\ell+1}) \\
f^*(x_1,\ldots,x_\ell) - y^* &= x_{\ell+1}q'_{\ell+1}(x_1,\ldots,x_{\ell+1}) + (r_y - r_f)x_{\ell+1} + \sum_{i=1}^{\ell}(x_i - t_i)q'_i(x_1,\ldots,x_{\ell+1}) \\
f^*(x_1,\ldots,x_\ell) - (r_y - r_f)x_{\ell+1} - y^* &= x_{\ell+1}q'_{\ell+1}(x_1,\ldots,x_{\ell+1}) + \sum_{i=1}^{\ell}(x_i - t_i)q'_i(x_1,\ldots,x_{\ell+1}).
\end{aligned}$$

Now let f' be the $\ell+1$ variable polynomial defined as $f'(x_1,\ldots,x_{\ell+1}) \stackrel{\text{def}}{=} f^*(x_1,\ldots,x_\ell) - (r_y - r_f)x_{\ell+1}$ and let $t' \in \mathbb{F}^{\ell+1}$ defined as $t' = (t^*_1,\ldots,t^*_\ell,0)$. From the above relation it follows that $f'(x_1,\ldots,x_{\ell+1}) - y^* = \sum_{i=1}^{\ell+1}(x_i - t'_i)q'_i(x_1,\ldots,x_{\ell+1})$, therefore t' is a root of the polynomial $f'' \stackrel{\text{def}}{=} f'(x_1,\ldots,x_{\ell+1}) - y^*$, i.e.,

$f''(t') = 0$ which implies that

$$\begin{aligned} f'(t_1, \ldots, t_{\ell+1}) - y^* &= 0 \\ f^*(t_1, \ldots, t_\ell) - (r_y - r_f) \cdot 0 - y^* &= 0 \\ f^*(t_1, \ldots, t_\ell) &= y^* \end{aligned}$$

which implies that y^* is the correct evaluation of f^* on t^*, i.e., $\delta = 0$. If that is the case, $\mathcal{B}$ has already aborted, therefore conditioned on not aborting this will never happen.

In all other cases, the polynomial division is possible therefore we can write

$$\begin{aligned} e(g,g)^{\delta} &= e(g,g)^{(s_1+\tau)Q(s_1)+R} \\ e(g,g)^{\frac{\delta}{s_1+\tau}} &= e(g,g)^{Q(s_1)+\frac{R}{s_1+\tau}} \\ e(g,g)^{\frac{\delta-R}{s_1+\tau}} &= e(g,g)^{Q(s_1)}. \end{aligned}$$

If $\delta = R$ (case (1) above), then it follows that $e(g,g)^0 = e(g,g)^{Q(s_1)}$, i.e., $s_1 = s$ is root of Q. Therefore, $s = y$ for some $y \in Y$ (and therefore $|Y| > 0$). Since factorization can be done in deterministic polynomial time $\mathcal{B}$ always succeeds in computing this y and $e(g,g)^{\frac{1}{y+\tau}} = e(g,g)^{\frac{1}{s+\tau}}$ thus $\mathcal{B}$ succeeds in breaking Assumption 1 in this case. If $\delta \neq R$ (case (2) above), from the above it holds that

$$\begin{aligned} e(g,g)^{\frac{\delta-R}{s_1+\tau}} &= e(g,g)^{Q(s_1)} \\ e(g,g)^{\frac{1}{s_1+\tau}} &= e(g,g)^{Q(s_1)\cdot(\delta-R)^{-1}} \end{aligned}$$

therefore, in this case too, $\mathcal{B}$ succeeds in breaking Assumption 1 in this case.

Since the two cases are complementary, $\mathcal{B}$ always succeeds, conditioned on not aborting. Thus, it remains to bound the probability of aborting. $\mathcal{B}$ can only abort in three cases. If extraction fails, if $y^* = f^*(t^*)$, or if $\tau = -y$. The former can only happen with negligible probability. This holds since, if verification succeeds it must be that $e(\mathsf{com}_{i,2}, g) = e(\mathsf{com}_{i,1}, g^{\alpha})$ for $i = 1, \ldots, \ell+1$ and by Assumption 2, extraction for any of $\mathcal{E}_1, \ldots, \mathcal{E}_{\ell+1}$ fails with negligible probability. Since ℓ is polynomial in λ it follows that the probability any of them fails (which by a union bound is at most equal to the sum of each individual failure probability) is also negligible. The second happens by assumption with probability at most $1-\epsilon$ (as $\mathcal{A}$ wins with probability at least ϵ), whereas the third happens with negligible probability $O(2^{-\lambda})$ as τ is chosen uniformly at random from $\mathbb{F}$. By a union bound, the abort probability is at most $(1-\epsilon)+\mathsf{neg}(\lambda)$. Thus the success probability of $\mathcal{B}$ is $\epsilon - \mathsf{neg}(\lambda)$ which is non-negligible as we assumed that ϵ is non-negligible. Since $\mathcal{B}$ succeeds in breaking Assumption 1 this contradicts our original assumption and our proof is complete.

Evaluation Extractability. This follows almost directly from soundness and polynomial extractability. In particular, let $\mathcal{A}$ be an PPT adversary that plays the evaluation extractability game. Let $\mathcal{A}_f$, $\mathcal{A}_y$ be two adversaries that on input the same input as $\mathcal{A}$, run $\mathcal{A}$'s code internally but only output com_f^*, com_y^* respectively and then halt. Clearly, both adversaries are PPT. Moreover, whenever $\mathsf{CheckCom}(\mathsf{com}_f^*, \mathsf{vp})$ and $\mathsf{Ver}(\mathsf{com}_f^*, t^*, \mathsf{com}_y^*, \pi^*, \mathsf{vp})$ output accept, it follows that: (1) by polynomial extractability there exist extractor $\mathcal{E}_f$ with access to the code and random tape of $\mathcal{A}_f$ that with all but negligible probability outputs f, r_f such that $\mathsf{CommitPoly}(f, r_f, \mathsf{pp}) = \mathsf{com}_f^*$, and (2) by Assumption 2, since Ver accepted (and recall that as a sub-routine, Ver checks that $e(\mathsf{com}_{y,1}, g^{\beta}) = e(\mathsf{com}_{y,2}, g)$) there exists PPT extractor with access to the code and random tape of $\mathcal{A}_y$ that with all but negligible probability, outputs $y, r_y \in \mathbb{F}$ such that $g^{y+r_y s_{\ell+1}} = \mathsf{com}_{y,1}$.

It remains to show that the event $E = \{f(t^*) \neq y$, where f is the output of $\mathcal{E}_f$ and y is the output of $\mathcal{E}_y\}$ occurs with negligible probability. For contradiction, assume $\Pr[E] = \epsilon$, for some non-negligible ϵ. Then we can build adversary $\mathcal{A}'$ that breaks the binding property of our scheme, as follows.

1. On input $(1^\lambda, \mathsf{pp})$, $\mathcal{A}'$ runs $\mathcal{A}$ internally and receives $(t^*, \pi^*, \mathsf{com}_f^*, \mathsf{com}_y^*)$.

2. $\mathcal{A}'$ runs $\mathcal{E}_f, \mathcal{E}_y$ on the same input as $\mathcal{A}$ to receive f, r_f, y, r_y.

3. $\mathcal{A}'$ outputs $(f, t^*, y, \pi^*, \mathsf{com}_f^*, \mathsf{com}_y^*, r_f, r_y)$ as a challenge for the soundness game.

$\mathcal{A}'$ is clearly PPT as $\mathcal{A}, \mathcal{E}_f, \mathcal{E}_y$ are all PPT. Note that whenever E occurs, $\mathcal{A}'$ wins. Assuming that $\Pr[E] = \epsilon$, it follows that $\mathcal{A}'$ breaks the binding property, for which we proved above that it can only happen with negligible probability. This concludes our proof.

Zero Knowledge. We build our simulator Sim that operates as follows.

1. On input $(1^\lambda, \ell, d)$, run $\mathsf{KeyGen}(1^\lambda, \ell, d)$ and receive pp, vp. Set $\sigma = \alpha, \beta, s_1, \ldots, s_{\ell+1}$. Choose $r_f \in \mathbb{F}$ uniformly at random and set $\mathsf{com}_f = (g^{r_f}, g^{\alpha \cdot r_f})$. Send $\mathsf{vp}, \mathsf{com}_f$ to $\mathcal{A}$.

2. Receive k from $\mathcal{A}$.

3. For $i = 1, \ldots, k$ repeat:

   (a) Receive t_i from $\mathcal{A}$.

   (b) Choose $r_{1i}, \ldots, r_{\ell i}, r_{yi} \in F$ uniformly at random.

   (c) Compute $\mathsf{com}_{yi} = (g^{r_{yi} s_{\ell+1}}, g^{\beta r_{yi} s_{\ell+1}})$, $\mathsf{com}_{ji} = (g^{r_{ji} s_{\ell+1}}, g^{\alpha r_{ji} s_{\ell+1}})$ for $j = 1, \ldots, \ell$ and $\mathsf{com}_{\ell+1 i} = (g^{r_f - r_{yi} - \sum_{j=1}^{\ell} r_{ji}(s_j - t_{ij})}, g^{\alpha(r_f - r_{yi} - \sum_{j=1}^{\ell} r_{ji}(s_j - t_{ij}))})$.

   (d) Output $(\mathsf{com}_{yi}, \pi_i = (\mathsf{com}_{1i}, \ldots, \mathsf{com}_{\ell+1 i}), \sigma)$.

Sim is clearly PPT as all the above steps can be computed in time polynomial in λ. Next, note that since r_f and $r_{1i}, \ldots, r_{\ell i}$, for all i, are chosen uniformly at random, it follows that $\mathsf{com}_f, \mathsf{com}_{1i}, \ldots, \mathsf{com}_{\ell i}$ are indistinguishable from uniformly chosen elements from $\mathbb{G}$. Moreover, this holds both in the real and the ideal game execution since in the former the discrete logs of these elements are computed as the sum of a polynomial evaluation in $\mathbb{F}$ and an element of $\mathbb{F}$ chosen uniformly at random. Finally, note that in both games, for any i, fixing $\mathsf{com}_f, \mathsf{com}_{1i}, \ldots, \mathsf{com}_{\ell i}$ also fixes a unique element $\mathsf{com}_{\ell+1 i} \in \mathbb{G}$. From the above, it follows that for any (even unbounded) adversary $\mathcal{A}$ and all $f \in \mathbb{F}$, it holds that the view from the execution of $\mathsf{Real}_{\mathcal{A},f}(1^\lambda)$ and $\mathsf{Ideal}_{\mathcal{A},\mathsf{Sim}}(1^\lambda)$ is indistinguishable, thus Construction 1 is perfect zero-knowledge.

□

4 A Sum-check Protocol over Homomorphic Commitments

Overview. The sum-check protocol [24] is an interactive protocol which allows a prover $\mathcal{P}$ to convince a verier $\mathcal{V}$ of the validity of statements of the form $H = \sum_{b_1 \in \{0,1\}} \cdots \sum_{b_\ell \in \{0,1\}} g(b_1, \cdots, b_\ell)$ where $g : \mathbb{F}^\ell \to \mathbb{F}$ is an ℓ-variate polynomial over some finite field $\mathbb{F}$. While $\mathcal{V}$ can directly compute H using $O(2^\ell)$ evaluation of g, the sum-check protocol reduces $\mathcal{V}$'s work to be polynomial in ℓ. Indeed, the protocol proceeds as follows (in ℓ rounds). In in the first round, $\mathcal{P}$ sends $\mathcal{V}$ the univariate polynomial $g_1(x) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_\ell \in \{0,1\}} g(x, b_2, \cdots, b_\ell)$. $\mathcal{V}$ then checks that the degree of g_1 is the same as the variable degree of x_1 in g, rejecting otherwise. $\mathcal{V}$ then proceeds in sending $\mathcal{P}$ a uniform challenge $r_1 \in \mathbb{F}$. Next, during the ith round of the sum-check protocol, $i = 2, \cdots, \ell$, $\mathcal{P}$ sends the univariate polynomial $g_i(x) = \sum_{b_{i+1} \in \{0,1\}} \cdots \sum_{b_\ell \in \{0,1\}} g(r_1, \cdots, r_{i-1}, x, b_{i+1}, \cdots, b_\ell)$. Upon receiving g_i, $\mathcal{V}$ checks that $g_{i-1}(r_{i-1}) = g_i(0) + g_i(1)$, rejecting otherwise. $\mathcal{V}$ then proceeds to send a uniform r_i to $\mathcal{P}$, which is the challenge to be used in the next round. Finally, at the last round, $\mathcal{V}$ accepts only if $g(r_1, \cdots, r_\ell) = g_\ell(r_\ell)$.

In the sequel, we define the degree of each monomial of g as the sum of the powers of its variables. We then define g's total degree as the maximal degree of any of its monomials. The following theorem is due to [24].

Construction 2 (Sum-check Protocol Over Homomorphic Commitment Schemes). *Let $\mathbb{F}$ be a prime-order finite field, and let λ be a security parameter. In addition, let $\mathcal{C}omm$ be a linearly homomorphic commitment scheme as described in Section 2.3, $\mathsf{cp} \leftarrow \mathcal{S}etup(1^\lambda)$, and let $g(b_1, \cdots, b_\ell)$ be an ℓ-variate total-degree-d polynomial over $\mathbb{F}$ which is represented using m coefficients $a_0, \cdots, a_m$. Consider the following protocol between $\mathcal{P}$ and $\mathcal{V}$ for convincing $\mathcal{V}$ that $t_0 = \sum_{b_1 \in \{0,1\}} \cdots \sum_{b_\ell \in \{0,1\}} g(b_1, \cdots, b_\ell)$ is a valid opening to some commitment com_0. That is, that he knows ρ_0 such that $\mathsf{com}_0 = \mathcal{C}om(\mathsf{cp}, t_0, \rho_0)$.*

1. *For all $i = 1, \cdots, \ell$ perform the following.*
   (a) *Define $g_i(x) = \sum_{b_{i+1}, \cdots, b_\ell \in \{0,1\}} g(r_1, \cdots, r_{i-1}, x, b_{i+1}, \cdots, b_\ell)$ and let $a_0, \cdots, a_m$ be the coefficients of g_i.*
   (b) *For every $0 \leq j \leq m$ $\mathcal{P}$ computes $\mathsf{com}_{a_j} \leftarrow \mathcal{C}om(\mathsf{cp}, a_j, \rho_{a_j})$ where $\rho_{a_j} \in \mathbb{F}$ is selected uniformly at random and sends $(\mathsf{com}_{a_0}, \cdots, \mathsf{com}_{a_m})$ to $\mathcal{V}$.*
   (c) *$\mathcal{V}$ computes $\mathsf{com}^*_{i-1} \leftarrow \mathsf{com}_{a_0} \cdot \prod_{k=0}^{m} \mathsf{com}_{a_k}$ which is a commitment to $g_i(0) + g_i(1)$.*
   (d) *$\mathcal{V}$ and $\mathcal{P}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, t_{i-1}, \rho_{i-1}, \rho_{a_0} + \sum_{j=0}^{m} \rho_{a_j}; \mathsf{com}_{i-1}, \mathsf{com}^*_{i-1})$.*
   (e) *$\mathcal{V}$ generates a random value r_i and sends it to $\mathcal{P}$.*
   (f) *Both $\mathcal{V}$ and $\mathcal{P}$ compute $\mathsf{com}_i \leftarrow \mathcal{E}val(\mathsf{cp}, \mathsf{com}_{a_0}, \cdots, \mathsf{com}_{a_m}, 1, r_i, \cdots, r_i^m)$.*
   (g) *$\mathcal{P}$ sets $t_i \leftarrow g_i(r_i)$ and $\rho_i \leftarrow \sum_{j=0}^{m} \rho_{a_j} r_i^j$*
2. *$\mathcal{V}$ computes $\mathsf{com}^*_\ell \leftarrow \mathcal{C}om(\mathsf{cp}, g(r_1, \cdots, r_\ell), \rho_{\ell+1})$ and sends com^*_ℓ and $\rho_{\ell+1}$ to $\mathcal{P}$.*
3. *Both $\mathcal{V}$ and $\mathcal{P}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, g(r_1, \cdots, r_\ell), \rho_\ell, \rho_{\ell+1}; \mathsf{com}_\ell, \mathsf{com}^*_\ell)$.*

Theorem 2. *For any ℓ-variate, total-degree-d polynomial $g : \mathbb{F}^\ell \to \mathbb{F}$, the sum-check protocol is an interactive proof for the (no-input) function $\sum_{b_1, \cdots, b_\ell \in \{0,1\}} g(b_1, \cdots, b_\ell)$ with soundness $d \cdot \ell / |\mathbb{F}|$. Moreover, $\mathcal{P}$ performs $O(2^\ell \cdot poly(\ell))$ arithmetic operations over $\mathbb{F}$ and $\mathcal{V}$ performs $poly(\ell)$ arithmetic operations over $\mathbb{F}$ and one evaluation of g on a random point r.*

The Sum-check Protocol over Homomorphic Commitments. Unfortunately the messages exchanged during the sum check protocol reveal the coefficients of $g_1, \cdots, g_\ell$, thus leaking additional information about the values of g, beyond H. This is problematic since we would like to use the sum-check protocol as part of a zero-knowledge argument system of NP, where leaking additional evaluations of g might leak information about the prover's witness. To that end, we execute the sum-check protocol over an additively homomorphic commitment scheme. In particular, we consider the Pedersen commitment scheme, modified as described in Section 2.3. The modified protocol starts by having $\mathcal{P}$ commit to the coefficients of g and by providing a commitment com_0 to the value H. Next, at round i, instead of having $\mathcal{P}$ send to $\mathcal{V}$ the coefficients of g_i, we modify the sum-check protocol and have $\mathcal{P}$ send commitments to these coefficients to $\mathcal{V}$. Since Pedersen commitments are linearly homomorphic, $\mathcal{V}$ can locally compute a commitment com^* to $g_i(0) + g_i(1)$ and then check (using the $\mathcal{ZK}_{eq}$ protocol) that com^* commits to the same value as com_{i-1}, thus verifying that indeed $g_{i-1}(r_{i-1}) = g_i(0) + g_i(1)$. In case this verification succeeds, $\mathcal{V}$ sends a uniformly random challenge r_i to $\mathcal{P}$, and both $\mathcal{P}$ and $\mathcal{V}$ use the homomorphic properties of the Pedersen commitment scheme in order to obtain a commitment com_i to the evaluation of g_i on r_i. $\mathcal{P}$ and $\mathcal{V}$ repeat the above, using com_i and r_i in round $i + 1$.

Formally, consider the protocol presented in Construction 2. We now state the following theorem (we are only interested in proving "regular" soundness and not knowledge soundness).

Theorem 3. *For any ℓ-variate, total-degree-d polynomial $g : \mathbb{F}^\ell \to \mathbb{F}$ with m non-zero coefficients, assuming $\mathcal{C}omm$ is a linearly homomorphic commitment scheme, as described in Section 2.3, and $\mathcal{ZK}_{eq}$ is a zero-knowledge non-interactive argument for testing equality of commitments for $\mathcal{C}omm$, Construction 2*

is an interactive argument with soundness $d \cdot \ell/|\mathbb{F}|$ for the following language

$$L = \left\{ (\mathsf{cp}, \mathsf{com}_0, g; \rho_0) : \mathsf{com}_0 \leftarrow \mathcal{C}om \left(\mathsf{cp}, \sum_{b_1, \cdots, b_\ell \in \{0,1\}} g(b_1, \cdots, b_\ell), \rho_0 \right) \text{ and } \mathsf{cp} \leftarrow \mathcal{S}etup(1^\lambda) \right\}$$

where λ is the security parameter.

Proof Sketch. The completeness property immediately follows from Construction 2. We now proceed to argue about the soundness property.

Soundness. Let $g(b_1, \cdots, b_\ell)$ be an ℓ-variate total-degree-d polynomial over a finite field $\mathbb{F}$. We begin by observing that the commitment com_0 and all coefficient commitments $\mathsf{com}_{a_j,i}$ for $i = 1, \ldots, \ell$, $j = 1, \ldots, m$ are extractable. That is, for each of them there exists a polynomial-time extractor that receives the same input as the adversary $\mathcal{A}$ and outputs with all but negligible probability a valid pre-image from $\mathbb{F}$, whenever $\mathcal{A}$ succeeds in convincing $\mathcal{V}$. This follows under Assumption 2 using the same argument as in the proof of Theorem 1 (recall that, as explained in Section 2.3, we implicitly assume that whenever $\mathcal{V}$ receives a commitment he checks whether it is well-formed and rejects otherwise).

Next, we distinguish between the following two complementary cases.

1. **There exists $0 \leq i \leq \ell$ such that the extracted pre-images for $\mathsf{com}_i, \mathsf{com}_i^*$ are not equal.** In this case, this directly contradicts the soundness of the $\mathcal{ZK}_{eq}$ protocol executed in Steps 1d and 3 of Construction 2. This means that $\mathcal{A}$ can be used to construct a black box adversary $\mathcal{A}'$ which breaks the soundness of the $\mathcal{ZK}_{eq}$ protocol.

2. **For all $0 \leq i \leq \ell$ it holds that the extracted pre-images for $\mathsf{com}_i, \mathsf{com}_i^*$ are equal.** In this case let t_0^* be the extracted pre-image of com_0, and let h_i^* be the extracted pre-image of com_i for all $i = 0, \cdots, \ell - 1$. Also, let g_i^* be the polynomial defined by coefficients $a_0, \cdots, a_m$ which are the pre-images extracted from the commitments $\mathsf{com}_{a_0}, \cdots, \mathsf{com}_{a_m}$ sent by $\mathcal{P}$ in Step 1b of Construction 2 during the $(i+1)$th round. Notice that since com_i and com_i^* have the same pre-image h^*, by construction of com_i^* is holds that $h_i^* = g_i^*(0) + g_i^*(1)$. Due to this, notice that $(t_0^*, g, (r_i, g_i^*, h_i^*)_{i=1,\cdots,\ell})$ is a valid transcript for a (possibly) cheating prover $\mathcal{P}^*$ controlled by $\mathcal{A}$ trying to convince a verifier $\mathcal{V}$ that indeed $t_0 = \sum_{b_1 \in \{0,1\}} \cdots \sum_{b_\ell \in \{0,1\}} g(b_1, \cdots, b_\ell)$.

   Thus, if $t_0 \neq \sum_{b_1 \in \{0,1\}, \ldots, b_\ell \{0,1\}} g(b_1 \ldots, b_\ell)$ then $\mathcal{A}$ can be used in a black-box manner in order to break the soundness property of the sum-check interactive proof protocol.

□

Moreover, we prove the following lemma that will be helpful for us while proving the zero-knowledge property of our argument.

Lemma 2. *For every verifier $\mathcal{V}^*$ and for every ℓ-variate, total-degree-d polynomial $g : \mathbb{F}^\ell \to \mathbb{F}$ with m non-zero coefficients, there exists a simulator Sim such that Sim is capable of simulating from $\mathsf{cp}, \mathsf{com}_0, m$ and t_0 (without using g) the partial view of $\mathcal{V}^*$ defined by $\mathsf{cp}, \mathsf{com}_0$ as well as the messages obtained during only Step 1 of Construction 2.*[1]

Proof Sketch. We build simulator Sim which simulates the view of $\mathcal{V}$ during Step 1 of Construction 2 as follows. First, Sim receives as input commitment parameters cp, commitment com_0, an upper bound m on the number of coefficients of g, as well as the value $t_0 = \sum_{b_1, \cdots, b_\ell \in \{0,1\}} g(b_1, \cdots, b_\ell)$. Sim proceeds to simulate Step 1 of Construction 2 as follows.

1. For $i = 1, \ldots, \ell$:

[1] Notice that the partial view does not include the coefficients $a_1, \cdots, a_m$ of g.

(a) Sim chooses coefficients $a_0, \ldots a_{m-1}$ chosen uniformly at random from $\mathbb{F}$ and sets a_m such that $t_{i-1} = a_0 + a_m + \sum_{j=0}^{m-1} a_j$. Let $g_i(x)$ be the polynomial denoted by coefficients $a_0, \ldots, a_m$ and notice that $t_{i-1} = g_i(0) + g_i(1)$.

(b) For every $0 \leq j \leq m$, Sim computes $\mathsf{com}_{a_j} \leftarrow \mathcal{C}om(\mathsf{cp}, a_j, \rho_{a_j})$ where $\rho_{a_j} \in \mathbb{F}$ is selected uniformly at random, and sends $(\mathsf{com}_{a_0}, \cdots, \mathsf{com}_{a_m})$ to $\mathcal{V}^*$.

(c) Sim computes $\mathsf{com}^*_{i-1} \leftarrow \mathsf{com}_{a_0} \cdot \prod_{k=0}^{m} \mathsf{com}_{a_k}$.

(d) Let Sim_{eq} be the simulator guaranteed from the zero knowledge property of $\mathcal{ZK}_{eq}$. Sim runs simulator Sim_{eq} on inputs $(\mathsf{cp}, \mathsf{com}^*_{i-1}, \mathsf{com}_{i-1})$ in order to simulate $\mathcal{V}^*$'s view during the execution of $\mathcal{ZK}_{eq}$ the ith round.

(e) Upon receiving r_i from $\mathcal{V}^*$, Sim computes $\mathsf{com}_i \leftarrow \mathcal{E}val(\mathsf{cp}, \mathsf{com}_{a_0}, \cdots, \mathsf{com}_{a_m}, 1, r_i, \cdots, r_i^m)$ and sets $t_i \leftarrow g_i(r_i)$ and $\rho_i \leftarrow \sum_{j=0}^{m} \rho_{a_j} r_i^j$.

The produced transcript is indistinguishable from the one $\mathcal{V}^*$ gets while interacting with $\mathcal{P}$ since: (i) the coefficients $a_0, \ldots, a_m$ for each round i satisfy the same relation with respect to t_{i-1} in both cases, (ii) $\mathcal{C}omm$ is statistically hiding, i.e., each commitment is indistinguishable from a commitment to a random value, and (iii) the output of Sim_{eq} for round i is indistinguishable from the messages received by $\mathcal{V}^*$ while running $\mathcal{ZK}_{eq}$ on the same values. In the following, we consider a slightly modified simulator Sim that outputs as secret state the values (t_ℓ, ρ_ℓ) to be used when building a larger simulator that runs Sim as a black box. □

5 A CMT Protocol over Homomorphic Commitments

Cormode et al.[12] present an efficient interactive proof protocol (to which we shall refer in the sequel as the CMT protocol) for a prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ of the validity of a statement of the form "$y = C(x)$" for some depth-d arithmetic circuit C and input x. The protocol proceeds in d rounds, processing one circuit layer at a time, from the output layer (layer 0) to the input layer (layer d). At round i, $\mathcal{P}$ reduces the claim about the values of C at the ith layer to a claim about the values of C's values in layer $i+1$. At the final round (round d), the protocol terminates with a claim about the input wires of C. Since the input x is known to $\mathcal{V}$, $\mathcal{V}$ directly checks the validity of this claim, rejecting otherwise.

Concretely (using the notation of Section 2.4), at the ith round (corresponding to the ith layer of C) $\mathcal{V}$ verifies that for every gate g in the ith layer of C, $V_i(g)$ correctly outputs the values of g. Next, since V_i already represents the values of g as summation of wiring predicates and gate values, $\mathcal{V}$ can verify the correct computation of ith layer, using the sum-check protocol from Section 4. However, since the soundness guarantee of the sum-check protocol depends on the size of the underlying field $\mathbb{F}$, we replace V_i (which is defined over $\mathsf{GF}(2)$) with its multilinear extension $\tilde{V}_i$ which is defined over a finite field $\mathbb{F}$ as

$$\tilde{V}_i(z) = \sum_{\substack{g\in\{0,1\}^{s_i} \\ u,v\in\{0,1\}^{s_{i+1}}}} f_{i,z}(g,u,v) \stackrel{\text{def}}{=} \sum_{\substack{g\in\{0,1\}^{s_i} \\ u,v\in\{0,1\}^{s_{i+1}}}} \tilde{\beta}_i(z,g) \cdot \Big(\tilde{\mathsf{add}}_{i+1}(g,u,v) \cdot (\tilde{V}_{i+1}(u) + \tilde{V}_{i+1}(v)) + \tilde{\mathsf{mult}}_{i+1}(g,u,v) \cdot (\tilde{V}_{i+1}(u) \cdot \tilde{V}_{i+1}(v)) \Big), \tag{1}$$

where $\tilde{\mathsf{add}}_i$ (resp., $\tilde{\mathsf{mult}}_i$) is the multilinear extension of add_i (resp., mult_i) and $\tilde{\beta}_i$ is the multilinear extension of the selector function that takes two s_i-bit inputs a, b and outputs 1 if $a = b$ and 0 else.

Assume that the output of C is a single value. At a high level, the protocol proceeds as follows. At the first round, $\mathcal{P}$ claims that $y = \tilde{V}_0(0)$ for some value y. $\mathcal{V}$ and $\mathcal{P}$ now execute the sum-check protocol, verifying that $\tilde{V}_0(r_0) = \sum_{\substack{g\in\{0,1\}^{s_0} \\ u,v\in\{0,1\}^{s_1}}} f_{0,r_0}(g,u,v)$ at a random point $r_0 \in \mathbb{F}^{s_0}$ which is generated by $\mathcal{V}$. At the last step of the sum-check protocol, the sum-check verifier $\mathcal{V}_{sc}$ needs to evaluate f_{0,r_0} at a random

point $\rho_0 \in \mathbb{F}^{s_0+2s_1}$ generated by $\mathcal{V}_{sc}$. Next, since f_{0,r_0} depends on $\tilde{V}_1$, in order to evaluate $f_{0,r_0}(\rho_0)$ it is the case that $\mathcal{V}_{sc}$ needs to evaluate $\tilde{V}_1(q_1)$ and $\tilde{V}_1(q_2)$ where $(q_1, q_2) \in \mathbb{F}^{2s_1}$ are the last $2s_1$ entries of ρ_0. However, since $\mathcal{V}$ does not have access to the correct gate values of layer-1 and thus cannot evaluate $\tilde{V}_1$ on his own, he relies on $\mathcal{P}$ to provide him with two values a_1, a_2 claiming that $a_1 = \tilde{V}_1(q_1)$ and $a_2 = \tilde{V}_1(q_2)$. Finally, $\mathcal{P}$ and $\mathcal{V}$ now again execute the sum-check protocol in order to verify the validity of the two claims made by $\mathcal{P}$' thus reducing $\mathcal{P}$'s claims regarding the correct computation of the ith layer of C to claims about correct computation of layer $i-1$. Applying this method repeatedly for d rounds, the final claim made by $\mathcal{P}$ is with regards to an evaluation of $\tilde{V}$ (the multilinear extension of the circuits inputs) which can be directly checked by $\mathcal{V}$ since he has access to x.

Unfortunately, utilizing the above-described approach directly leads to the number of claims being verified by $\mathcal{V}$ to double with each circuit layer. For a depth-d circuit, this results in a total of 2^d executions of the sum-check protocol. We now describe the method for condensing the CMT protocol to use a single claim to be verified by $\mathcal{V}$ per layer. Indeed, instead of having $\mathcal{V}$ verify $\tilde{V}_1(q_1)$ and $\tilde{V}_1(q_2)$ directly, let $\gamma : \mathbb{F} \to \mathbb{F}^{s_1}$ be the unique line such that $\gamma(0) = q_1$ and $\gamma(1) = q_2$. $\mathcal{P}$ then sends $\mathcal{V}$ the degree-s_1 polynomial $h(x) = \tilde{V}_1(\gamma(x))$. $\mathcal{V}$ then checks that $h(0) = a_1$ and that $h(1) = a_2$. Next, $\mathcal{V}$ generates a random point r and both $\mathcal{P}$ and $\mathcal{V}$ perform a single execution of the sum-check protocol in order to verify that $h(r_1) = \tilde{V}_1(\gamma(r))$. Thus, this procedure reduces the total number of the invocations of the sum-check protocol from $O(2^d)$ to $O(d)$.

Formally, consider the following theorem.

Theorem 4 ([18, 12])**.** *Let $C : \mathbb{F}^n \to \mathbb{F}^k$ be a depth-d layered arithmetic circuit over a finite field $\mathbb{F}$. The CMT protocol described above is an interactive proof for the function computed by C with soundness $O(d \cdot \log S/|\mathbb{F}|)$, where S is the maximal number of gates per circuit layer. Moreover, $\mathcal{P}$'s running time is $O(|C| \log S)$ and the protocol uses $O(d \log S)$ rounds of interaction. Finally, in case $\tilde{\mathsf{add}}_i$ and $\tilde{\mathsf{mult}}_i$ are computable in time $O(\text{polylog}\, S)$ for all the layers of C, then the running time of the verifier $\mathcal{V}$ is $O(n + k + d \cdot \text{polylog}\, S)$.*

The CMT Protocol over Homomorphic Commitments. Similarly to the case of the standard sum-check protocol, the messages exchanged during the CMT execution leak information about the intermediate values of the circuit C and thus potentially about the circuit's input (which in our case will include the prover's witness). Thus, similarly to Section 4, we execute the CMT protocol over homomorphic commitments and use the commitment's hiding property to conceal from $\mathcal{V}$ information regarding the circuit's internal wires.

The modified protocol proceed as follows. First, $\mathcal{P}$ sends to $\mathcal{V}$ commitments $\mathsf{com}_{x_1}, \ldots, \mathsf{com}_{x_n}$ to the n inputs of C and a commitment com_0 to 1 (the claimed output of the circuit when evaluated on x). Next, at round-i, both $\mathcal{P}$ and $\mathcal{V}$ use the first step of the sum-check protocol over homomorphic commitments, resulting in $\mathcal{V}$ obtaining a commitment com'_i on a_i which is claimed by $\mathcal{P}$ to be equal to $\tilde{V}_{i-1}(r'_i)$, where r'_i is uniformly generated by $\mathcal{V}$. Let q_1, q_2 be the last $2s_i$ elements of r'_i. $\mathcal{P}$ then computes $t_1 = \tilde{V}_i(q_1)$, $t_2 = \tilde{V}_i(q_2)$, $t_3 = t_1 \cdot t_2$ and commits to t_1, t_2, t_3 resulting in $\mathsf{com}_{t_1}, \mathsf{com}_{t_2}, \mathsf{com}_{t_3}$. $\mathcal{V}$ then verifies (using the $\mathcal{ZK}_{prod}$ protocol) that indeed com_{t_3} is a commitment to the multiplication of t_1 and t_2 and uses the homomorphic properties of the commitment scheme and the $\mathcal{ZK}_{eq}$ protocol in order to check that the value com'_i provided earlier by $\mathcal{P}$ is indeed a commitment to the evaluation of $\tilde{V}_{i-1}(r'_i)$. Both $\mathcal{V}$ and $\mathcal{P}$ then use the homomorphic properties of the commitment scheme and the $\mathcal{ZK}_{eq}$ protocol in order for $\mathcal{V}$ to obtain a commitment com_i to a value $a_i = \tilde{V}_i(\gamma(r''_i))$, where r''_i is generated by $\mathcal{V}$, and proceed to the next round of the protocol using a_i and random point $r_i = \gamma(r''_i)$. Finally $\mathcal{P}$ reveals x and r_0 to $\mathcal{V}$ who then checks their consistency with the initial commitments, evaluates the polynomial $\tilde{V}_x$ on the last random point established r_d and both parties use $\mathcal{ZK}_{eq}$ to establish that a commitment to this evaluation has the same pre-image as com_d. We stress that this entire last step (which clearly would violate any notion of zero-knowledge) will not be a step of our final construction; it will instead by replaced with appropriate evocations of zk-VPD.

Formally, consider the protocol presented in Construction 3 and the following theorem.

Construction 3 (CMT Protocol Over Homomorphic Commitment Schemes). *Let $\mathbb{F}$ be a prime-order finite field, and let λ be a security parameter, $\mathcal{C}omm$ be a linearly homomorphic commitment scheme as described in Section 2.3, and let $\mathsf{cp} \leftarrow \mathcal{S}etup(1^\lambda)$. In addition, let $C : \mathbb{F}^n \to \mathbb{F}$ be a depth-d layered arithmetic circuit and let $x \in \mathbb{F}^n$ be inputs of C such that $C(x) = 1$. Consider the following protocol between $\mathcal{P}$ and $\mathcal{V}$ for convincing $\mathcal{V}$ that x is a valid opening to a series of commitments $\mathsf{com}_{x_i} \leftarrow \mathcal{C}om(\mathsf{cp}, x_i, \rho_{x_i})$, where $x_i \in \mathbb{F}$ is the i-th element of input x.*

1. *Both parties set $a_0 = 1$ and $r_0 = 0$. $\mathcal{P}$ generates ρ_0 uniformly at random, computes $\mathsf{com}_0 \leftarrow \mathcal{C}omm(\mathsf{cp}, a_0, \rho_0)$ and sends it to $\mathcal{V}$.*
2. *For all $i = 1, \cdots, d$ perform the following.*
   (a) *$\mathcal{V}$ and $\mathcal{P}$ execute Step 1 of Construction 2 on input a_{i-1}, polynomial $\tilde{V}_{i-1}$ (as per Equation 1), and randomness ρ_{i-1} for $\mathcal{P}$ and $\mathsf{com}_{i-1}, r_{i-1}$ for $\mathcal{V}$. As a result, $\mathcal{V}$ obtains a commitment $\mathsf{com}'_i = \mathcal{C}om(t, \rho_i)$ (where t and ρ_i are known to $\mathcal{P}$ and not to $\mathcal{V}$) where $\mathcal{P}$ claims that $t = \tilde{V}_{i-1}(r'_i)$ with r'_i having been selected uniformly at random by $\mathcal{V}$.*
   (b) *Let (q_1, q_2) be the last $2s_i$ elements of r'_i. $\mathcal{P}$ computes $t_1 = \tilde{V}_i(q_1)$, $t_2 = \tilde{V}_i(q_2)$, and $t_3 = t_1 \cdot t_2$. $\mathcal{P}$ then computes commitments $\mathsf{com}_{t_1} \leftarrow \mathcal{C}om(\mathsf{cp}, t_1, \rho_{t_1})$, $\mathsf{com}_{t_2} \leftarrow \mathcal{C}om(\mathsf{cp}, t_2, \rho_{t_2})$, and $\mathsf{com}_{t_3} \leftarrow \mathcal{C}om(\mathsf{cp}, t_3, \rho_{t_3})$ which he sends to $\mathcal{V}$.*
   (c) *$\mathcal{V}$ and $\mathcal{P}$ perform $\mathcal{ZK}_{prod}(\mathsf{cp}, t_1, t_2, t_3, \rho_{t_1}, \rho_{t_2}, \rho_{t_3}; \mathsf{com}_{t_1}, \mathsf{com}_{t_2}, \mathsf{com}_{t_3})$.*
   (d) *Using Equation 1 $\mathcal{V}$ can express $\tilde{V}_{i-1}(r'_i)$ as a linear function of $r'_i, \tilde{V}_i(q_1), \tilde{V}_i(q_2), \tilde{V}_i(q_1) \cdot \tilde{V}_i(q_2)$. Thus, using $\mathcal{E}val$, $\mathcal{V}$ can obtain a new commitment com^*_i to the evaluation of $\tilde{V}_{i-1}(r'_i)$ and let ρ^*_i be the corresponding randomness.*
   (e) *$\mathcal{V}$ and $\mathcal{P}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, \tilde{V}_{i-1}(r'_i), \rho_i, \rho^*_i; \mathsf{com}'_i, \mathsf{com}^*_i)$.*
   (f) *Let $\gamma : \mathbb{F} \to \mathbb{F}^{s_i}$ be the line defined by $\gamma(0) = q_1$ and $\gamma(1) = q_2$ and let $h(x)$ be the degree-s_i polynomial such that $h(x) = \tilde{V}_i(\gamma(x))$ and $h_0, \ldots, h_{s_i}$ be its coefficients. For $j = 0, \ldots, s_i$, $\mathcal{P}$ computes commitments $\mathsf{com}_{h_j} \leftarrow \mathcal{C}om(\mathsf{cp}, h_j, \rho_{h_j})$ and sends them to $\mathcal{V}$.*
   (g) *$\mathcal{V}$ computes $\mathsf{com}_{h(0)} \leftarrow \mathsf{com}_{h_0}$ and $\mathsf{com}_{h(1)} \leftarrow \prod_{j=0}^{s_i} \mathsf{com}_{h_j}$ which are commitments to $h(0)$ and $h(1)$ respectively.*
   (h) *$\mathcal{V}$ and $\mathcal{P}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, t_1, \rho_{t_1}, \rho_{h_0}; \mathsf{com}_{t_1}, \mathsf{com}_{h(0)})$ and $\mathcal{ZK}_{eq}(\mathsf{cp}, t_2, \rho_{t_2}, \sum_{j=0}^{s_i} \rho_{h_j}; \mathsf{com}_{t_2}, \mathsf{com}_{h(1)})$.*
   (i) *$\mathcal{V}$ chooses $r''_i \in \mathbb{F}$ uniformly at random, sets $r_i \leftarrow \gamma(r''_i)$ and sets $\mathsf{com}_i \leftarrow \mathcal{E}val(\mathsf{cp}, \mathsf{com}_{h_0}, \cdots, \mathsf{com}_{h_{s_i}}, 1, r''_i, \cdots, r''^{s_i})$.*
   (j) *$\mathcal{V}$ sends r''_i and com_i to $\mathcal{P}$. $\mathcal{P}$ sets $r_i \leftarrow \gamma(r''_i)$, $a_i \leftarrow \tilde{V}_i(\gamma(r''_i))$ and $\rho_i \leftarrow \sum_{j=0}^{s_i} r''^j \rho_{h_j}$.*
3. *$\mathcal{P}$ sends to $\mathcal{V}$ the input x and randomness ρ_0 and ρ_{x_i} for $1 \le i \le n$. $\mathcal{V}$ verifies that $\mathsf{com}_{x_i} = \mathcal{C}om(\mathsf{cp}, x_i, \rho_{x_i})$ for $1 \le i \le n$ and that $\mathsf{com}_0 = \mathcal{C}om(\mathsf{cp}, 1, \rho_0)$.*
4. *Let $\tilde{V}_x$ be the multilinear extension of the polynomial V_x satisfying $V_x(i) = x_i$ for all $i = 1, \cdots, n$. The verifier computes $\mathsf{com}^*_x \leftarrow \mathcal{C}om(\mathsf{cp}, \tilde{V}_x(r_d), \rho^*_d)$ where ρ^*_d is chosen uniformly at random.*
5. *$\mathcal{V}$ and $\mathcal{P}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, \tilde{V}_x(r_d), \rho_d, \rho_x; \mathsf{com}^*_x, \mathsf{com}_d)$. $\mathcal{V}$ accepts if the protocol accepts and rejects otherwise.*

Theorem 5. *Let $C : \mathbb{F}^n \to \mathbb{F}^k$ be a depth-d layered arithmetic circuit over a finite field $\mathbb{F}$. Assuming $\mathcal{C}omm$ is an linearly homomorphic commitment scheme as described in Section 2.3, $\mathcal{ZK}_{eq}$ is a zero-knowledge argument for testing equality of committed values, and $\mathcal{ZK}_{prod}$ is a zero-knowledge argument for testing the product relation between three commitments in $\mathcal{C}omm$, the CMT protocol presented in Construction 3 is an interactive argument with soundness $O(d \cdot \log S/|\mathbb{F}|)$ for for the following relation*

$$\mathcal{R} = \left\{ \left((\mathsf{cp}, C, \mathsf{com}_{x_1}, \ldots, \mathsf{com}_{x_n}); (x_1, \cdots, x_n, \rho_{x_1}, \ldots, \rho_{x_n}) \right) : C(x_1, \cdots, x_n) = 1 \wedge \bigwedge_{i=1}^{n} \mathsf{com}_{x_i} = \mathcal{C}om(\mathsf{cp}, x_i, \rho_{x_i}) \right\}$$

where $\mathsf{cp} \leftarrow \mathcal{S}etup(1^\lambda)$, λ is the security parameter, n is the input size of C, and S is the maximal number

of gates per circuit layer in C.

Proof Sketch. The completeness property immediately follows from Construction 3. We now proceed to argue about the soundness property.

Soundness. Soundness follows by a similar argument as in Theorem 3. Indeed, let $\mathcal{P}^*$ be an cheating prover which convinces $\mathcal{V}$ (with non-negligible probability) of a claim "$1 = C(x)$" for some x and C, such that $1 \neq C(x)$. Using Assumption 2, for each commitment that $\mathcal{V}$ receives from $\mathcal{P}^*$, there exists a polynomial-time extractor with access to $\mathcal{P}^*$'s code and random tape that outputs (with all but negligible probability) a corresponding commitment pre-image.

Next, we define the following events.

1. Event A takes place if the extracted pre-image for com_d is not equal to $\tilde{V}_x(r_d)$ or there exists $1 \leq i \leq d$ such that the extracted pre-images for $\mathsf{com}'_i, \mathsf{com}^*_i$ during Step 2e, or the extracted pre-images for $\mathsf{com}_{t_1}, \mathsf{com}_{h(0)}$ or $\mathsf{com}_{t_2}, \mathsf{com}_{h(1)}$ during Step 2h are not equal.

2. Event B takes place if there exists $1 \leq i \leq d$ such that the extracted pre-image for com_{t_3} is not equal to the product of the extracted pre-images for $\mathsf{com}_{t_1}, \mathsf{com}_{t_2}$ during Step 2c

3. Event C_i (for $0 \leq i \leq d$) takes place if $\tilde{V}_i(r_i) = a_i$ (where $\tilde{V}_i$ is as defined in Equation 1) when evaluating C on the extracted pre-image of com_x and a_i is the extracted pre-image of com_i.

Note that assuming $C(x) \neq 1$ is equivalent to assuming $\tilde{V}_0(r_0) \neq a_0$ i.e., that $\neg C_0$ occurred. Next, we study the following (exhaustive) cases.

- **Event A occurs.** We argue about this case in exactly the same manner as in the proof of Theorem 3. That is, this directly contradicts the soundness of the $\mathcal{ZK}_{eq}$ protocol executed in Steps 5, 2e, or 2h of Construction 2 since $\mathcal{P}^*$ can be used in a black box manner to construct an adversary $\mathcal{A}$ which breaks the soundness of the $\mathcal{ZK}_{eq}$ protocol.

- **Event B occurs.** Again, this directly contradicts the soundness of the $\mathcal{ZK}_{prod}$ protocol executed in Step 2c of Construction 2 since $\mathcal{P}^*$ can thus be used in a black box manner to construct an adversary $\mathcal{A}$ which breaks the soundness of the $\mathcal{ZK}_{prod}$ protocol.

- **There exist $1 \leq i \leq d$ such that $\neg C_{i-1}$ occurs and events A, B do not.** We will now prove that this case contradicts the soundness property of Construction 2. Note that since $\mathcal{V}$ computes $\tilde{V}_x(r_d) \stackrel{\text{def}}{=} \tilde{V}_d(r_d)$ himself and since event A did not occur, C_d must occur. Therefore in this case it holds $\neg A \wedge \neg B \wedge \neg C_i \wedge \neg C_0 \wedge C_d$. Therefore, there must exist i' such that $\neg C_{i'-1} \wedge C_{i'}$.

  Let $a^*_{i'-1}$ be the extracted pre-image of $\mathsf{com}_{i'-1}$, $a^*_{i'}$ be the extracted pre-image of $\mathsf{com}_{i'}$. Since $\neg C_{i'-1} \wedge C_{i'}$ holds we obtain that $a^*_{i'} = \tilde{V}_{i'}(r_i)$ and that $a^*_{i'-1} \neq \tilde{V}_{i'-1}(r_{i'-1})$. Next, since $a^*_{i'} = \tilde{V}_{i'}(r_i)$ from Steps 2f-2h we obtain that (except with negligible probability) it holds that the extracted pre-images of $\mathsf{com}_{t_1}, \mathsf{com}_{t_2}$ are indeed equal to $\tilde{V}_{i'}(q_1), \tilde{V}_{i'}(q_2)$ (where (q_1, q_2) are the last $2s_i$ elements of r'_i).

  We now claim that $\mathcal{P}^*$ can be used in black-box manner to construct an adversary $\mathcal{A}$ that succeeds in falsely proving that $a^*_{i'-1}$ is equal to $\tilde{V}_{i'-1}(r_{i'-1})$, thus contradicting the soundness of Construction 2. Indeed, let $\mathcal{V}_{sum-check}$ be a verifier for Construction 3 using the coefficients of $\tilde{V}_{i'-1}$. $\mathcal{A}$ then performs (using the code of $\mathcal{P}^*$) Step 1 of Construction 2. Since the verifier $\mathcal{V}$ for Construction 3 did not reject while interacting with $\mathcal{P}^*$ (and in particular, did not reject during Step 2a with $i = i'$), $\mathcal{V}_{sum-check}$ will not reject as well. Notice that, at this point in Construction 2, it is the case that com_ℓ is a commitment to $\tilde{V}_{i'-1}(r'_i)$ and so is $\mathsf{com}'_{i'}$ (defined in Step 2a of Construction 3 with $i = i'$).

  Next, $\mathcal{V}_{sum-check}$ and $\mathcal{A}$ proceed by performing Steps 2 and 3 of Construction 2. We now argue that $\mathcal{V}_{sum-check}$ will not reject during Step 3 of Construction 2. Indeed, since the extracted pre-images

t_1, t_2 of $\mathsf{com}_{t_1}, \mathsf{com}_{t_2}$ are equal to $\tilde{V}_{i'}(q_1), \tilde{V}_{i'}(q_2)$ and since $\mathcal{V}$ did not reject during Step 2c and 2d of Construction 3 with $i = i'$, we obtain that $\mathcal{V}$ successfully performed a step which is equivalent to Step 2 of Construction 2. Thus, $\mathcal{V}$ holds a commitment $\mathsf{com}^*_{i'}$ to $\tilde{V}_{i'-1}(r'_i)$ (using the notation of Construction 3) and $\mathcal{V}_{sum-check}$ holds a commitment com^*_ℓ to the same value $\tilde{V}_{i'-1}(r'_i)$. At this point, ass explained above we also have that that the pre-images of com_ℓ and $\mathsf{com}'_{i'}$ are both equal to $\tilde{V}_{i'-1}(r'_i)$ as well. Since event A did not occur, it holds that the pre-images of $\mathsf{com}'_{i'}$ and $\mathsf{com}^*_{i'}$ also have the same pre-image. By transitivity, we obtain that com^*_ℓ is a commitment to the same value as com_ℓ, thus $\mathcal{V}_{sum-check}$ will not reject during Step 3 of Construction 2. Therefore, we have violated the soundness of Construction 2 by allowing $\mathcal{A}$ to falsely prove that $a^*_{i'-1} = \tilde{V}_{i'-1}(r_{i'-1})$. □

Moreover, we prove the following lemma that will be helpful for us while proving the zero-knowledge property of our argument.

Lemma 3. *For every verifier $\mathcal{V}^*$ and for every depth-d layered circuit $C : \mathbb{F}^n \to \mathbb{F}^k$ over a finite field $\mathbb{F}$ there exists a simulator Sim such that Sim is capable of simulating the view of $\mathcal{V}^*$ in steps 1 and 2 of Construction 3 from C, without access to x.*

Proof Sketch. We build simulator Sim that simulates the view of $\mathcal{V}$ during Steps 1 and 2 of Construction 3. The simulator gets as input commitment parameters cp and a circuit C and proceeds as follows.

1. Sim sets $a_0 = 1$ and $r_0 = 0$, and computes $\mathsf{com}_x \leftarrow \mathcal{C}om(\mathsf{cp}, 0, \rho_x)$ for some ρ_x generated at random. Sim then sends com_x to $\mathcal{V}^*$.

2. Sim generates ρ_0 uniformly at random, computes $\mathsf{com}_0 \leftarrow \mathcal{C}om(\mathsf{cp}, a_0, \rho_0)$ and sends it to $\mathcal{V}$.

3. Sim proceeds to simulate Step 2 of Construction 3 as follows. For all $i = 1, \cdots, d$ Sim performs the following.

   (a) Let $\mathsf{Sim}_{sum-check}$ be the simulator from the proof of Lemma 2. Sim runs $\mathsf{Sim}_{sum-check}$ on input $(\mathsf{cp}, \mathsf{com}_{i-1}, s_{i-1}, a_{i-1})$, in order to simulate $\mathcal{V}^*$'s view during the execution of Step 2a. In addition to the final message com'_i sent to $\mathcal{V}^*$, $\mathsf{Sim}_{sum-check}$ also outputs a secret state (a_i, ρ_i) which is not forwarded to $\mathcal{V}^*$. Notice that a_i is the simulated value of $\tilde{V}_{i-1}(r'_i)$ where r'_i was chosen by $\mathcal{V}^*$.

   (b) Let (q_1, q_2) be the last $2s_i$ elements of r'_i. Sim chooses simulated values $t_1, t_2 \in \mathbb{F}$ for $\tilde{V}_i(q_1)$ and $\tilde{V}_i(q_2)$ such that a_i (which is the simulated value $\tilde{V}_{i-1}(r'_i)$), t_1, t_2 (which are the simulated values for $\tilde{V}_i(q_1)$ and $\tilde{V}_i(q_2)$) and r'_i satisfy Equation 1.

   (c) Sim then computes $\mathsf{com}_{t_j} \leftarrow \mathcal{C}om(\mathsf{cp}, t_j, \rho_{t_j})$ for $j = 1, 2, 3$, where $t_3 = t_1 \cdot t_2$ and $\rho_{t_1}, \rho_{t_2}, \rho_{t_3}$ are chosen uniformly at random from $\mathbb{F}$, and forwards them to $\mathcal{V}^*$.

   (d) Let Sim_{prod} be the simulator guaranteed from the zero knowledge property of $\mathcal{ZK}_{prod}$. Sim runs simulator Sim_{prod} on input $\mathsf{cp}, \mathsf{com}_{t_1}, \mathsf{com}_{t_2}, \mathsf{com}_{t_3}$ in order to simulate $\mathcal{V}^*$'s view during the execution of Step 2c of Construction 3).

   (e) Sim performs Step 2d of Construction 3 using the values r'_i, t_1, t_2, t_3. This results in a commitment com^*_i to the value of $\tilde{V}_{i-1}(r'_i)$.

   (f) Sim runs simulator Sim_{eq} on input $\mathsf{cp}, \mathsf{com}'_i, \mathsf{com}^*_i$ in order to simulate $\mathcal{V}^*$'s view during the execution of Step 2e of Construction 3).

   (g) Sim computes $\mathsf{com}_{h(0)}$ as a fresh commitment to t_1. For $j = 1, \ldots, s_i - 1$, Sim chooses values $h_j \in \mathbb{F}$ uniformly at random. Moreover, he chooses h_{s_i} such that $\sum_{j=1}^{s_i} h_j + t_1 = t_2$ and for $j = 1, \ldots, s_i$ he computes $\mathsf{com}_{h_j} \leftarrow \mathcal{C}om(\mathsf{cp}, h_j, \rho_{h_j})$. Sim then sends $\mathsf{com}_{h_0}, \ldots, \mathsf{com}_{h_{s_i}}$ to $\mathcal{V}^*$.

   (h) Sim computes $\mathsf{com}_{h(1)} \leftarrow \mathsf{com}_{h(0)} \cdot \prod_{j=1}^{s_i} \mathsf{com}_{h_j}$.

   (i) Sim runs simulator Sim_{eq} on input $\mathsf{cp}, \mathsf{com}_{t_1}, \mathsf{com}_{h(0)}$ and on input $\mathsf{cp}, \mathsf{com}_{t_2}, \mathsf{com}_{h(1)}$ in order to simulate $\mathcal{V}^*$'s view during the execution of Step 2h of Construction 3).

(j) Finally, Sim sets $r_i \leftarrow \gamma(r_i'')$ (where r_i'' was sent by $\mathcal{V}^*$ in Step 2j of Construction 3) and $a_i \leftarrow H(r_i'')$ (where H is the degree-s_i polynomial that has as coefficients $t_1 = h_0, h_1, \ldots, h_{s_i}$) and $\rho_i \leftarrow \sum_{j=0}^{s_i} r''^j \rho_{h_j}$. Finally, Sim computes $\mathsf{com}_i \leftarrow \mathcal{C}omm(a_i, \rho_i)$.

We claim that the view of $\mathcal{V}^*$ while interacting with Sim (for Steps 1,2 of Construction 2) is indistinguishable from the view he gets while interacting with the honest prover $\mathcal{P}$ since: (i) All triplets a_i and t_1, t_2 (for each round i) chosen by Sim satisfy Equation 1, (ii) All values h_j (for each round i) satisfy the condition $h_0 = t_1$ and $\sum_{j=1}^{s_i} h_j + t_1 = t_2$, (iii) by assumption, the messages received by $\mathcal{V}^*$ by $\mathsf{Sim}_{eq}, \mathsf{Sim}_{prod}, \mathsf{Sim}_{sum-check}$ (forwarded via Sim) are indistinguishable form the ones received while running $\mathcal{ZK}_{eq}, \mathcal{ZK}_{prod}$ and Construction 2 with the honest prover, and (iv) (ii) $\mathcal{C}omm$ is statistically hiding. □

6 A Zero-knowledge Argument with Function Independent Preprocessing

In this section we construct our zero knowledge proof system with function independent preprocessing. At a high level, similarly to [30], we observe that it is possible to use the CMT protocol in order to verify the correct evaluation of a circuit C on input x and a witness w assuming that $\mathcal{V}$ can somehow evaluate a specific polynomial P (which only depends on x and w but not on C) on a random point r generated by $\mathcal{V}$. Adopting the approach of [30], we observe that the evaluation of P can be done using a verifiable polynomial evaluation protocol (VPD). Since the CMT protocol does not require any preprocessing, the only part of our construction where preprocessing is required is the VPD protocol. Next, since the VPD protocol is only used to verify the input layer of the circuit, our preprocessing depends only on (an upper bound on) the length of the circuit's input and witness, and not on the circuit's size of specific wiring pattern.

While the above-outlined construction does produce an argument with function independent preprocessing, the messages exchanged during the CMT and VPD protocol leak information about the values of the internal wires of C and thus can potentially leak information about the witness w. We remove such leakdge, by running the CMT protocol over homomorphic commitments, as described in Sections 4 and 5, and by replacing the VPD construction of [30] with our zk-VPD construction from Section 3. Formally, consider the protocol presented in Construction 4 and the following theorem.

Theorem 6. *For any circuit size parameter t, input size n and finite field $\mathbb{F}$, Construction 4 is a zero-knowledge argument system for the relation*

$$\mathcal{R} = \{(C, x; w) : C \in \mathcal{C}_{\mathbb{F}} \wedge |C| \leq t \wedge \mathsf{inp}(C) \leq n \wedge C(x; w) = 1\}.$$

Moreover, for every $(C, x; w) \in R$ the running time of $\mathcal{P}$ is $O(|C| \cdot \log(\mathsf{width}(C)))$ and if C is log-space uniform then the running time of $\mathcal{V}$ is $O(|x| + d \cdot \mathrm{poylog}(|C|))$. Finally $\mathcal{P}$ and $\mathcal{V}$ interact $O(d \log(\mathsf{width}(C)))$ rounds where d is the depth of C. In case d is polylog $(|C|)$*, the above construction is a* succinct *argument.*

Proof Sketch. The completeness property immediately follows from Construction 4. We now proceed to argue about the knowledge soundness property.

Knowledge Soundness. Let $\mathcal{A}$ be a reduced version of $\mathcal{P}^*$ that aborts right after outputting $\mathsf{com}_{\tilde{V}_d}$. By the polynomial extractability property of Construction 1, there exists extractor $\mathcal{E}'$ that upon the same input as $\mathcal{P}^*$ and the same random tape, outputs a n-variable degree-variable 1, polynomial f and randomness ρ_f such that $\mathsf{CommitPoly}(f, \rho_f, \mathsf{pp}) = \mathsf{com}_{\tilde{V}_d}$ with all but negligible probability. We are now ready to build our extractor $\mathcal{E}$ as follows:

1. Run $\mathcal{E}'(1^\lambda, \mathsf{pp})$ and receive polynomial f and randomness ρ_f. If f is not a n-variable degree-variable 1, polynomial f, abort.

Construction 4 (Zero-knowledge Delegation Protocol). *Let λ be a security parameter and let $\mathbb{F}$ be a prime order field such that $|\mathbb{F}|$ is exponential in λ. In additional, let n be an input size parameter and let t be a circuit size parameter. In the following, for simplicity of exposition we assume that n is a power of 2. Consider the algorithms $\mathcal{G}, \mathcal{P}, \mathcal{V}$ described below.*

- ***Preprocessing Phase.*** *The parameter generator $\mathcal{G}$ on input $1^n, 1^t, 1^\lambda$ runs $(\mathsf{pp}, \mathsf{vp}) \leftarrow \mathsf{KeyGen}(1^\lambda, n, 1)$. The proving key pk is set to be pp and the verification key vk is set to be vp.*
- ***Evaluation Phase.*** *Let $C : \mathbb{F}^{n_x+n_w} \to \mathbb{F}$ be a depth-d layered arithmetic circuit over $\mathbb{F}$ such that $|C| \leq t$ and $n_x + n_w \leq n$. In addition, let $x \in \mathbb{F}^{n_x}$ and $w \in \mathbb{F}^{n_w}$ such that $C(x; w) = 1$. Assume that $n_w/n_x = 2^m - 1$ for some $m \in \mathbb{N}$. Consider the following protocol between $\mathcal{P}$ and $\mathcal{V}$.*
  1. *Let $\tilde{V}_d$ be the multilinear extension of the input layer of C evaluated on $(x; w)$. $\mathcal{P}$ commits to the values of $\tilde{V}_d$ by executing $\mathsf{com}_{\tilde{V}_d} \leftarrow \mathsf{CommitPoly}(\tilde{V}_d, \rho_{\tilde{V}_d}, \mathsf{pp})$ where $\rho_{\tilde{V}_d}$ is generated uniformly at random. $\mathcal{P}$ then sends $\mathsf{com}_{\tilde{V}_d}$ to $\mathcal{V}$.*
  2. *$\mathcal{V}$ runs $\mathsf{CheckCom}(\mathsf{com}_{\tilde{V}_d}, \mathsf{vp})$. In case $\mathsf{CheckCom}$ rejects, $\mathcal{V}$ rejects as well.*
  3. *$\mathcal{V}$ and $\mathcal{P}$ execute Steps 1 and 2 of Construction 3. In case Construction 3 rejects, so does $\mathcal{V}$. Otherwise, at the end of Step 2 of Construction 3 $\mathcal{V}$ holds a commitment com_d of an evaluation of $\tilde{V}_d$ at a random point r_d chosen by $\mathcal{V}$ while $\mathcal{P}$ holds the randomness ρ_d used to generate com_d.*
  4. *$\mathcal{P}$ executes $(\mathsf{com}^*_d, \pi) \leftarrow \mathsf{CommitValue}(\tilde{V}_d, r_d, \tilde{V}_d(r_d), \rho_{\tilde{V}_d}, \rho_{\tilde{V}_d(r_d)}, \mathsf{pp})$ where $\rho_{\tilde{V}_d(r_d)}$ is generated uniformly at random and sends (com^*_d, π) to $\mathcal{V}$.*
  5. *Upon receiving (com^*_d, π), $\mathcal{V}$ executes $\mathsf{Ver}(\mathsf{com}^*_d, r_d, \mathsf{com}_{\tilde{V}_d}, \pi, \mathsf{vp})$. In case Ver rejects, so does $\mathcal{V}$.*
  6. *$\mathcal{P}$ and $\mathcal{V}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, \tilde{V}_d(r_d), \rho_d, \rho_{\tilde{V}_d(r_d)}; \mathsf{com}^*_d, \mathsf{com}_d)$. (Note that cp is a subset of vp.) In case $\mathcal{ZK}_{eq}$ rejects so does $\mathcal{V}$.*
  7. *$\mathcal{V}$ computes the multilinear extension $\tilde{x}$ of the input x, generates a random point $r_x \in (\mathbb{F}^{\log n_x} \times 0^{\log n_w})$ and sends r to $\mathcal{P}$.*
  8. *Upon receiving r_x, $\mathcal{P}$ executes $(\mathsf{com}^*_x, \pi_x) \leftarrow \mathsf{CommitValue}(\tilde{V}_d, r_x, r_{\tilde{V}_d}, \rho_{r_x}, \mathsf{pp})$ where ρ_{r_x} is generated uniformly at random and sends $(\mathsf{com}^*_x, \pi_x)$ to $\mathcal{V}$. Next, $\mathcal{V}$ executes $\mathsf{Ver}(\mathsf{com}^*_x, r_x, \mathsf{com}_{\tilde{V}_d}, \pi_x, \mathsf{vp})$. In case Ver rejects, so does $\mathcal{V}$.*
  9. *$\mathcal{V}$ computes $\mathsf{com}_x \leftarrow \mathcal{C}om(\mathsf{vp}, \tilde{x}(r'_x), \rho'_x)$ where ρ'_x is generated uniformly at random and r'_x is defined to be the first $\log n_x$ elements of r_x. $\mathcal{V}$ sends ρ'_x to $\mathcal{P}$.*
  10. *Both $\mathcal{P}$ and $\mathcal{V}$ perform $\mathcal{ZK}_{eq}(\mathsf{cp}, \tilde{V}_d(r_x), \rho_{r_x}, \rho'_x; \mathsf{com}_x, \mathsf{com}^*_x)$. In case $\mathcal{ZK}_{eq}$ rejects so does $\mathcal{V}$.*

2. Output $w = (f(n_x), \ldots, f(n_w - 1))$.

We now argue that assuming $\mathcal{P}^*$ successfully convinced a verifier $\mathcal{V}$, it is indeed the case that $C(x, w) = 1$.

First, notice that com^*_x (produced via $\mathsf{CommitValue}$) and com_x are of the same format, i.e., regular Pedersen commitments under the same cp parameters (as described in Section 2.3). Thus, by the soundness property of the $\mathcal{ZK}_{eq}$ protocol we obtain that com_x and com^*_x are commitments to the same pre-image. Next, let $\mathcal{E}_{vpd,x}$ be the extractor for $\mathcal{P}^*$ (limited to Steps 1 and 8 of Construction 4) guaranteed by the evaluation extractability of property of Construction 1, as per Definition 3. Since $\mathcal{P}^*$ convinces $\mathcal{V}$ we obtain that $\mathcal{E}_{vpd,x}$ on the same inputs as $\mathcal{P}^*$ outputs $f(r_x)$ as the pre-image of com^*_x (with high probability).

We now argue that $(f(0), \cdots, f(n_x - 1)) = x$. Indeed, notice that f is an n-variate variable-degree-1 polynomial and it is thus a multilinear extension. In addition, by construction of $\tilde{x}$ (Step 7 of Construction 4) it holds that $(\tilde{x}(0), \cdots, \tilde{x}(n_x - 1)) = x$. Next, since com_x and com^*_x are commitments to the same value, we obtain that $\tilde{x}(r'_x) = f(r_x)$. Thus, by the properties of multilnear extensions we obtain that with high probability it holds that $(f(0), \cdots, f(n_x - 1)) = (\tilde{x}(0), \cdots, \tilde{x}(n_x - 1)) = x$.

We now proceed to argue that $C(x, w) = 1$. Let $x' = (x, w)$. We now show how to construct a prover $\mathcal{P}^*_{cmt}$ which will convince a verifier $\mathcal{V}_{cmt}$ from Construction 3 that $C(x') = 1$. Using the soundness

property of Construction 3 we shall obtain that $C(x,w) = C(x') = 1$ with high probability. Indeed, let $x' = (x'_1, \cdots, x'_n)$, $\mathcal{P}^*_{cmt}$ starts by computing $\mathsf{com}_{x'_i} \leftarrow \mathcal{C}om(\mathsf{cp}, x'_i, \rho_{x'_i})$ where $\rho_{x'_i}$ is generated uniformly at random and cp was given to $\mathcal{P}^*$ by the parameter generator $\mathcal{G}$. $\mathcal{P}^*_{cmt}$ then sends $\mathsf{com}_{x'_1}, \cdots, \mathsf{com}_{x'_n}$ to $\mathcal{V}_{cmt}$ and proceeds as follows.

1. $\mathcal{P}^*_{cmt}$ sets $a_0 = 1, r_0 = 0$, generates ρ_0 uniformly at random, computes $\mathsf{com}_0 \leftarrow \mathcal{C}omm(\mathsf{cp}, a_0, \rho_0)$ and sends it to $\mathcal{V}_{cmt}$. $\mathcal{P}^*_{cmt}$ then emulates $\mathcal{G}$ and runs P^* until Step 3 of Construction 4, discarding messages sent to $\mathcal{V}_{cmt}$.

2. Using $\mathcal{P}$ (restricted to Step 3 of Construction 4), $\mathcal{P}^*_{cmt}$ now interacts with $\mathcal{V}_{cmt}$ during Step 2 of Construction 3 by forwarding messages between $\mathcal{V}_{cmt}$ and $\mathcal{P}$. At the end of this step $\mathcal{P}^*_{cmt}$ and $\mathcal{V}_{cmt}$ hold a commitment com_d and a random point r_d chosen by $\mathcal{V}^*_{cmt}$.

3. $\mathcal{P}^*_{cmt}$ then sends x' and the randomness $\rho_{x'_1}, \cdots, \rho_{x'_n}$ to $\mathcal{V}_{cmt}$.

4. $\mathcal{P}^*_{cmt}$ then runs P^* until Step 6 of Construction 4 again discarding messages sent to $\mathcal{V}_{cmt}$.

5. Using $\mathcal{P}$ (restricted to Step 6 of Construction 4), $\mathcal{P}^*_{cmt}$ now interacts with $\mathcal{V}_{cmt}$ during Step 5 of Construction 3 by forwarding messages between $\mathcal{V}_{cmt}$ and $\mathcal{P}$.

We now proceed to argue that since $\mathcal{P}$ convinces $\mathcal{V}$ it is the case that $\mathcal{P}^*_{cmt}$ convinces $\mathcal{V}_{cmt}$. Indeed, first notice that since Step 2 of Construction 4 involve running running Steps 1 and 2 of Construction 3 and since $\mathcal{V}$ did not reject we have that $\mathcal{V}^*_{cmt}$ will not reject as well. Next, since the commitments $\mathsf{com}_{x'_1}, \cdots, \mathsf{com}_{x'_n}$ to $\mathcal{V}_{cmt}$ sent by $\mathcal{P}^*_{cmt}$ to $\mathcal{V}_{cmt}$ are honestly computed commitments to the values of x' using the randomness $\rho_{x'_1}, \cdots, \rho_{x'_n}$, we obtain that $\mathcal{V}^*_{cmt}$ will not reject during Step 3 of Construction 3. It remains to show that $\mathcal{V}^*_{cmt}$ will not reject during Step 5 of Construction 3.

Indeed, notice that f is the unique multilinear extension of $x' = (x, w)$. Thus we have that the polynomial $\tilde{V}_x$ defined in Step 4 of Construction 3 actually equals f. Let $\mathcal{E}_{vpd,d}$ be the extractor for $\mathcal{P}^*$ (limited to Steps 1 and 4 of Construction 4) guaranteed by the evaluation extractability of property of Construction 1, as per Definition 3. Since $\mathcal{P}$ convinces $\mathcal{V}$ we have that with high probability $\mathcal{E}_{vpd,d}$ on the same inputs as $\mathcal{P}^*$ outputs $f(r_d)$ as the pre-image of com^*_d. Next, by uniqueness property of multilinear extensions, we have that the multilinear extension $\tilde{V}_{x'}$ of x' computed in Step 2 of Construction 3 equals f. This implies that commitment $\mathsf{com}^*_{x'}$ computed in Step 4 of Construction 3 (executed on input x') is also to a commitment to $\tilde{V}_{x'}(r_d) = f(r_d)$. Overall, since com_d is produced the same way in Construction 3 and Construction 4, we obtain that the $\mathcal{ZK}_{eq}$ protocol is executed on commitments to the same values. Thus, if $\mathcal{P}$ convinces $\mathcal{V}$ we obtain that $\mathcal{V}_{cmt}$ will also be convinced by $\mathcal{P}^*_{cmt}$.

Zero Knowledge. Let Sim_{vpd} be the simulator from Theorem 1 and Sim_{cmt} be the simulator from Lemma 3. Consider the simulator Sim which is defined as follows.

1. On input $(1^\lambda, C, x)$, Sim runs Sim_{vpd} on input $(1^\ell, n, 1)$ where n is the input size of C and receives commitment $\mathsf{com}_{\tilde{V}_d}$, parameters pp, vp, and state σ. Note that pp contains commitment parameters cp for the Pedersen commitment scheme, as defined in Section 2.3. Sim sends vp to $\mathcal{V}^*$.

2. Sim runs Sim_{cmt} on input (cp, C), in order to simulate $\mathcal{V}^*$'s view during the execution of Step 3 of Construction 4. Let com_d be the corresponding output forwarded to $\mathcal{V}^*$ and r_d be the last random point chosen by $\mathcal{V}^*$.

3. In order to simulate $\mathcal{V}^*$'s view during Step 4 of Construction 4, Sim runs Sim_{vpd} on input $(r_d, \sigma, \mathsf{pp})$ and receives commitment com^*_d, proof π, and new state σ. Sim then forwards (com^*_d, π) to $\mathcal{V}^*$.

4. Sim runs simulator Sim_{eq} on input $\mathsf{com}_d, \mathsf{com}^*_d$ in order to simulate $\mathcal{V}^*$'s view during the execution of Step 6 of Construction 4.

5. Upon receiving r_x from $\mathcal{V}^*$, Sim simulates $\mathcal{V}^*$'s view during Step 8 of Construction 4. To that end, Sim runs Sim_{vpd} on input $(r_x, \sigma, \mathsf{pp})$ and receives commitment com_x^*, proof π_x, and new state σ. Sim then forwards $(\mathsf{com}_x^*, \pi_x)$ to $\mathcal{V}^*$.

6. Upon receiving ρ'_x from $\mathcal{V}^*$, Sim computes $\mathsf{com}_x \leftarrow \mathcal{C}omm(\mathsf{vp}, \tilde{x}(r_x), \rho'_x)$. Next, Sim runs simulator Sim_{eq} on input $\mathsf{com}_x, \mathsf{com}_x^*$ in order to simulate $\mathcal{V}^*$'s view during the execution of Step 10 of Construction 4.

We claim that the view of $\mathcal{V}^*$ while interacting with Sim is indistinguishable from the view he gets while interacting with the honest prover $\mathcal{P}$ since: (i) $\mathcal{C}omm$ is statistically hiding, (ii) the messages received by $\mathcal{V}^*$ by Sim_{eq} (forwarded via Sim) are indistinguishable from the ones received while running $\mathcal{ZK}_{eq}$ with the honest prover, (iii) the messages received by $\mathcal{V}^*$ by Sim_{cmt} (forwarded via Sim) are indistinguishable from the ones received while running Construction 3 with the honest prover, and (iv) the messages received by $\mathcal{V}^*$ by Sim_{vpd} (forwarded via Sim) are indistinguishable from the ones received while running Construction 1 with the honest prover. Note that the values of commitments $\mathsf{com}_d, \mathsf{com}_d^*, \mathsf{com}_x, \mathsf{com}_x^*$ are independent of each other (modulo the common commitment parameters cp) in both the real and the ideal execution. In particular, the messages exchanged during Step 3 of Construction 1 do not depend on the value of com_x ($\mathsf{com}_{\tilde{V}_d}$ in the real execution). □

Asymptotic Complexity. Firstly, we note that Pedersen commitments, as well as protocols $\mathcal{ZK}_{eq}$, $\mathcal{ZK}_{prod}$, require a constant number of exponentiations and field operations (when instantiated as explained in Section 2.3). Then, the analysis of the asymptotic complexity of our argument follows in a straight forward manner from: (i) the analysis of CMT [12], (ii) the analysis of the standard VPD [30], (iii) the fact that the zk-VPD protocol of Section 3 has the same asymptotic behavior as the plain VPD of [30].

Acknowledgments

This work was supported in part by NSF awards #1514261 and #1526950, financial assistance award 70NANB15H328 from the U.S. Department of Commerce, National Institute of Standards and Technology, the 2017-2018 Rothschild Postdoctoral Fellowship, and the Defense Advanced Research Project Agency (DARPA) under Contract #FA8650-16-C-7622.

References

[1] E. Ben-Sasson, I. Bentov, A. Chiesa, A. Gabizon, D. Genkin, M. Hamilis, E. Pergament, M. Riabzev, M. Silberstein, E. Tromer, and M. Virza. Computational integrity with a public random string from quasi-linear PCPs. In *Advances in Cryptology—Eurocrypt 2017*, pages 551–579, 2017.

[2] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In *CRYPTO 2013*, pages 90–108. 2013.

[3] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Scalable zero knowledge via cycles of elliptic curves. In *CRYPTO 2014*, pages 276–294. 2014.

[4] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a von Neumann architecture. In *USENIX Security 2014*, 2014.

[5] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *ITCS 2012*, pages 326–349, 2012.

[6] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 326–349, 2012.

[7] N. Bitansky, R. Canetti, O. Paneth, and A. Rosen. On the existence of extractable one-way functions. STOC 2014, pages 505–514.

[8] D. Boneh and X. Boyen. Short signatures without random oracles. In *EUROCRYPT 2004*, pages 56–73, 2004.

[9] J. Bootle, A. Cerulli, P. Chaidos, and J. Groth. Efficient zero-knowledge proof systems. In *Foundations of Security Analysis and Design VIII - FOSAD 2014/2015/2016 Tutorial Lectures*, pages 1–31, 2016.

[10] E. Boyle and R. Pass. Limits of extractability assumptions with distributional auxiliary input. In *ASIACRYPT 2015*, pages 236–261.

[11] A. Chiesa, M. A. Forbes, and N. Spooner. A zero knowledge sumcheck and its applications. Cryptology ePrint Archive, Report 2017/305, 2017. `http://eprint.iacr.org/2017/305`.

[12] G. Cormode, M. Mitzenmacher, and J. Thaler. Practical verified computation with streaming interactive proofs. In *ITCS 2012*, pages 90–112, 2012.

[13] C. Costello, C. Fournet, J. Howell, M. Kohlweiss, B. Kreuter, M. Naehrig, B. Parno, and S. Zahur. Geppetto: Versatile verifiable computation. In *S&P 2015*, pages 253–270, 2015.

[14] R. Cramer and I. Damgård. Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free? In *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, pages 424–441, 1998.

[15] D. Fiore, C. Fournet, E. Ghosh, M. Kohlweiss, O. Ohrimenko, and B. Parno. Hash first, argue later: Adaptive verifiable computations on outsourced data. Cryptology ePrint Archive, 2016.

[16] J. A. Garay, P. D. MacKenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. *J. Cryptology*, 19(2):169–209, 2006.

[17] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct nizks without pcps. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 626–645, 2013.

[18] S. Goldwasser, Y. T. Kalai, and G. Rothblum. Delegating computation: interactive proofs for muggles. In *STOC 2008*, pages 113–122, 2008.

[19] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *STOC 1985*, pages 291–304.

[20] V. Goyal. Reducing trust in the PKG in identity based cryptosystems. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, pages 430–447, 2007.

[21] J. Groth. Short pairing-based non-interactive zero-knowledge arguments. In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, pages 321–340, 2010.

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

[23] J. Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 723–732, 1992.

[24] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[25] C. Papamanthou, E. Shi, and R. Tamassia. Signatures of correct computation. In *TCC 2013*, pages 222–242, 2013.

[26] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *S&P 2013*, pages 238–252, 2013.

[27] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, pages 129–140, 1991.

[28] R. S. Wahby, I. Tzialla, abhi shelat, J. Thaler, and M. Walfish. Doubly-efficient zkSNARKs without trusted setup.

[29] M. Walfish and A. J. Blumberg. Verifying computations without reexecuting them. *Commun. ACM*, 58(2):74–84, 2015.

[30] Y. Zhang, D. Genkin, J. Katz, D. Papadopoulos, and C. Papamanthou. vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases. In *IEEE Symposium on Security and Privacy (S&P) 2017*, 2017.