# Spartan and Bulletproofs are simulation-extractable (for free!)

Quang Dao[1] and Paul Grubbs[2]

[1] Carnegie Mellon University
qvd@andrew.cmu.edu **
[2] University of Michigan
paulgrub@umich.edu

**Abstract.** Increasing deployment of advanced zero-knowledge proof systems, especially zkSNARKs, has raised critical questions about their security against real-world attacks. Two classes of attacks of concern in practice are *adaptive soundness* attacks, where an attacker can prove false statements by choosing its public input after generating a proof, and *malleability* attacks, where an attacker can use a valid proof to create another valid proof it could not have created itself. Prior work has shown that *simulation-extractability* (SIM-EXT), a strong notion of security for proof systems, rules out these attacks.

In this paper, we prove that two transparent, discrete-log-based zkSNARKs, Spartan and Bulletproofs, are simulation-extractable (SIM-EXT) in the random oracle model if the discrete logarithm assumption holds in the underlying group. Since these assumptions are required to prove standard security properties for Spartan and Bulletproofs, our results show that SIM-EXT is, surprisingly, "for free" with these schemes. Our result is the first SIM-EXT proof for Spartan and encompasses both linear- and sublinear-verifier variants. Our result for Bulletproofs encompasses both the aggregate range proof and arithmetic circuit variants, and is the first to not rely on the algebraic group model (AGM), resolving an open question posed by Ganesh et al. (EUROCRYPT '22). As part of our analysis, we develop a generalization of the tree-builder extraction theorem of Attema et al. (TCC '22), which may be of independent interest.

## 1 Introduction

Zero-knowledge succinct non-interactive arguments of knowledge (zkSNARKs) allow a computationally-bounded prover to produce a proof about a NP statement *without* revealing anything other than its validity, and with proof size *sublinear* in the size of the witness [12,32,34]. An important line of recent works [7,9,13,15,17,20,32,35,38,44,48,57,60] has produced concretely efficient constructions of zkSNARKs for range proofs (e.g., Bulletproofs [16]) and general arithmetic circuit satisfiability (e.g., Spartan [54]) that have seen widespread deployment, especially in blockchains and cryptocurrencies [8,56,1,53,51,21,61], along with potential deployment in other areas of interests [40].

** Part of the work was done while the first author was at the University of Michigan.

As zkSNARKs are deployed in practice, it is important to understand whether they are actually secure against the kinds of attacks they are likely to face in real systems. Two security properties in particular give us pause: first, *adaptive soundness*, where a malicious prover must be unable to prove false statements even if it chooses the input after generating a proof; a related notion, adaptive knowledge soundness, guarantees extraction is possible against such an adaptive prover. The second property is *non-malleability*, where an accepting proof cannot be modified into a different one without knowing the witness. Neither property is implied by standard security definitions like non-adaptive (knowledge) soundness and zero knowledge, and schemes lacking these properties have been attacked in practice. For example, the voting system Helios was broken by an adaptive soundness attack on a zero-knowledge proof [11]; subsequent work found similar issues with the SwissPost voting system [41] for government elections. Though not against zero-knowledge proofs directly, malleability attacks are common in cryptocurrencies: for example, a malleability attack was allegedly used[3] to steal hundreds of millions of dollars from MtGox [47].

Fortunately, a security property called *simulation extractability* (SIM-EXT) implies adaptive (knowledge) soundness and non-malleability for zkSNARKs. Intuitively, SIM-EXT requires that the knowledge extractor succeeds even when the malicious prover can request *simulated* proofs for *arbitrary* statements. If we could prove zkSNARKs that are already used (or are likely to be used) in practice are SIM-EXT, we could be more confident they would resist advanced attacks that use adaptivity or malleability. Ideally, we could prove SIM-EXT in idealized models (e.g, the random oracle model, or ROM) and using assumptions (e.g. discrete-log), which are sufficient to prove standard security guarantees for zkSNARKs; this would indicate SIM-EXT comes (roughly) "for free".

A pair [29,30] of beautiful recent works by Ganesh et al. on SIM-EXT for zkSNARKs lays a path towards this goal. In [29], the authors give a general SIM-EXT theorem for zkSNARKs with updatable SRS, and use it to show PlonK [28], Marlin [20], and Sonic [45] are all SIM-EXT. In [30], the authors show SIM-EXT for Bulletproofs. Unfortunately, these works do not get us all the way towards our goal: first, because their techniques do not extend to *transparent* zkSNARKs like Spartan, which use different building blocks; second, because their results rely on the algebraic group model (AGM) [27] and are not currently known to hold from discrete log in the ROM.

### 1.1 Our Results

In this paper we prove that Spartan and Bulletproofs, two state-of-the-art transparent zkSNARKs, satisfy SIM-EXT in the ROM assuming only that the discrete log assumption holds. Our analyses required developing some new technical tools which may be of independent interest. Since Spartan and Bulletproofs were originally analyzed in the ROM and rely on the discrete log assumption, our

[3] A later study [22] cast some doubt on these claims, but did find evidence that over three hundred thousand Bitcoins had been involved in malleability attacks.

results imply these protocols are SIM-EXT "for free"—unmodified and without additional assumptions or stronger idealized models. More precisely, we prove SIM-EXT for two variants of Spartan—Spartan-NIZK, which has linear verifier time, and Spartan-SNARK, which has sublinear verifier time—instantiated with the default Hyrax-based polynomial commitment scheme [57]. These are the first proofs of SIM-EXT for any Spartan variant; we believe the Spartan-SNARK result is also the first proof of SIM-EXT for *any* transparent zkSNARK with sublinear verifier time. Similarly, we prove SIM-EXT for two versions of Bulletproofs—the aggregate range proof protocol BP-ARP used in several cryptocurrencies [36,46] and the arithmetic circuit satisfiability proof BP-ACSPf. Our proofs for these protocols are the first that do not rely on the algebraic group model.

Our results help to build confidence that state-of-the-art and deployed zkSNARKs resist the kinds of attacks these protocols will face as they see wider deployment in the future. Of more theoretical interest, they also imply the surprising fact that, in the ROM, a powerful primitive like a SIM-EXT zkSNARK can be built from a very weak assumption like discrete log.

The proofs of these four theorems are nontrivial; to prove them we built several new technical tools that may be of independent interest for future SIM-EXT analyses. We extended prior security notions for SIM-EXT to the transparent NIZK setting. We also needed to develop a nontrivial generalization of the tree extractor of Attema et al. [2].

Our analyses are also done with an emphasis on concrete security. Where possible we try to explicitly measure adversarial runtime and success probability. We also evaluate our bounds to estimate bit security for typical parameters for Spartan and Bulletproofs, and compare the bit security we obtain against other analyses where possible. Our bounds inherit the non-tightness common to most rewinding-based knowledge soundness analyses of NIZKs, and so the provable SIM-EXT security we get (in terms of bits) is quite low. Nevertheless, we believe our results can be improved by future work, and hope they eventually inform future parameter selection processes for zkSNARK standards [62].

### 1.2 Technical Overview

We follow the high-level approach to proving SIM-EXT developed by [24] and further generalized in [29,30]: for a Fiat-Shamir-compiled argument $\Pi_{\mathsf{FS}}$, SIM-EXT is implied by three other properties: (1) adaptive knowledge soundness, (2) a form of zero knowledge, and (3) a unique-response property. Since the results in [24] are specific to $\Sigma$-protocols and those in [30] are specific to the AGM, we take the SIM-EXT theorem of [29] as our starting point. After suitable adaptations to the transparent setting—we give these in Section 3—this theorem says that $\Pi_{\mathsf{FS}}$ is SIM-EXT in the ROM if:

1. it is adaptively knowledge sound (hereafter we will omit "adaptive" if it is clear from context),
2. it is perfect $k$-ZK, meaning that there exists a simulator that perfectly simulates honest proofs, but only programs the RO when generating the $k$-th challenge,

3. it is $k$-$\mathsf{UR}$ for the *same* round $k$, meaning no adversary can produce two accepting proofs that are identical up to the $k$-th round, *even if* it can program that round's challenge.

Proving these three properties is challenging, and required us to develop novel techniques which we summarize below.

*Knowledge Soundness.* We prove knowledge soundness for non-interactive versions of Spartan and Bulletproofs using a standard chain of reductions: namely, we reduce to the *special soundness* of the underlying interactive argument. Intuitively, special soundness of a proof system refers to the ability of an extractor to extract a witness from a *tree* of accepting transcripts with suitable structure. For multi-round protocols, special soundness is parameterized by a vector $(n_1, n_2, \ldots, n_r)$ describing the needed structure: each node at level one must have $n_1$ outgoing edges, level two nodes have $n_2$ edges, etc. Recently, Attema et al. [2] proved that knowledge soundness of the Fiat-Shamir-compiled argument $\Pi_{\mathsf{FS}}$ follows from special soundness of $\Pi$. We take it as our starting point; unfortunately, we cannot apply it directly to either Spartan or Bulletproofs. There are two main reasons for this: first, Attema et al. only consider perfect special soundness, but both Spartan and Bulletproofs only satisfy *computational* special soundness—roughly, because an extractor could fail to extract a witness from a tree of transcripts if a malicious prover finds a nontrivial discrete log relation.

The second reason is more subtle, and has to do with ensuring the tree has the right structure for extraction to be possible. In Attema et al., each node of the transcript tree is a prover message whose outgoing edges are labeled with distinct verifier challenges. For certain rounds in both Spartan and Bulletproofs, these verifier challenges must satisfy an extra predicate (beyond distinctness) for extraction to be possible. The tree-builder by Attema et al. does not support outputting such trees with extra structure.

To address these limitations, in Section 4 we give a generalization of Attema et al.'s tree-builder that has the desired properties. Our generalization captures other predicates on verifier challenges using the notion of an *efficiently-decidable partition* of the space of challenges. Intuitively, we build a wrapper algorithm that that sits between the prover and the Attema et al. tree-builder, and ensures the tree has the right structure by enforcing a partition of the challenge space.

Armed with this generalization, we prove computational special soundness for all variants of Spartan and Bulletproofs, which in turn implies knowledge soundness for their Fiat-Shamir-compiled versions. In both cases, our generalized tree-builder is a crucial component: for example, special soundness of Bulletproofs requires verifier challenges to be distinct modulo $\pm 1$, and Spartan requires linear independence for batching challenges sent during the sumcheck subprotocol.

*Building $k$-ZK Simulators.* For SIM-EXT, we must prove that Spartan and Bulletproofs are perfect $k$-ZK, meaning their proofs can be simulated by a simulator that can only program the RO in a single round. This is a departure from the

typical way to build NIZK simulators, which typically reprogram the RO in every round; in particular, doing this for Spartan and Bulletproofs requires giving entirely new simulators for these constructions.

We build our $k$-ZK simulator for Bulletproofs using an approach similar to [29]. Our $k$-ZK simulator construction for Spartan-NIZK uses a novel strategy that is worth highlighting here: it delays the round at which the RO is reprogrammed as late as possible in the protocol (in fact, our simulator only needs to reprogram the very last verifier challenge). Another interesting aspect of our $k$-ZK simulator for Spartan is that the same simulator works for both Spartan-NIZK and Spartan-SNARK—though the two protocols have major differences, we observe that the parts of Spartan-SNARK that work differently than Spartan-NIZK consist entirely of evaluating (extensions of) public matrices at a public point, and so are trivially simulatable.

*$k$-Unique Response.* To finish, we need to show Spartan and Bulletproofs are $k$-UR for the same $k$ as their respective $k$-ZK simulators. For Spartan variants, this is straightforward—we need only reprogram the RO during the final $\Sigma$-subprotocol, and it is well known [24] that $\Sigma$-protocols satisfy unique response.

For BP-ARP and BP-ACSPf, proving $k$-UR is more challenging. Indeed, prior work relied heavily on the AGM for analyzing unique response—for example, [30] observe that proving their version of unique response is the only part of their analysis that seems to actually rely on the AGM, and [29] need the AGM to show that KZG polynomial commitments are unique response.

We prove $k$-UR for Bulletproofs using a new proof strategy that, intuitively, replaces the AGM with extraction. In more detail, we extract witnesses from both proofs output by the $k$-UR adversary, then argue that *either* the witnesses are the same *or* the adversary has found a discrete log relation. To finish, we use the (novel) result that the Bulletproofs inner-product argument has unique proofs. Thus, if the witnesses are the same, the proofs must be the same as well.

*Limitations and open questions.* Our results do have some important limitations. Notably, our emphasis on removing the AGM means that the tightness of our Bulletproofs results is worse than the comparable result of [30]. While this is inherent in some sense because our extractors use rewinding instead of straight-line extraction, it means that the bit security of Bulletproofs and Spartan we could prove with typical parameters would come out to be quite poor. We discuss this in Section 7.

An interesting open problem we leave to future work is generalizing our techniques to other transparent zkSNARKs. In particular, there is a great deal of commonality between our proofs for Spartan and Bulletproofs which could be abstracted out and proven more generally. As many later works [44,60,35,55] have built on Spartan viewed as a *polynomial IOP* [17,20], it would be interesting to generalize our analyses into a SIM-EXT framework for polynomial IOPs.

### 1.3 Related Work

Simulation-extractability (SIM-EXT) for NIZKs was first defined in [52] (using different terminology). Thereafter, a long line of work refined and studied SIM-EXT [24,49], built SIM-EXT NIZKs [37], and showed that SIM-EXT is sufficient for other primitives like signatures of knowledge [19]. Other concurrent works attacked security of NIZKs in deployed systems, such as the voting system Helios, showing the importance of adaptive soundness [11] which is implied by SIM-EXT. Other work has looked at UC security for NIZKs [18] and given results on SIM-EXT in the QROM [23]. These works are not relevant to our results, since SIM-EXT does not imply UC security in the ROM; further, we study zkSNARKs built from discrete log, which is broken by quantum attacks.

The simulation-extractability of zkSNARKs is comparatively less well-studied. Two important prior works [29,30] which rely on the algebraic group model [27] (AGM) are described above; [30] proves SIM-EXT of Bulletproofs, and [29] proves SIM-EXT of Plonk [28], Marlin [20], and Sonic [45].

Other work has investigated generic transforms for achieving SIM-EXT from any zkSNARK [5], particularly focused on SIM-EXT transforms for the Groth16 zkSNARK [3,4]. Since Groth16 [38] is built using a different approach than either Spartan or Bulletproofs, and relies on non-falsifiable knowledge assumptions or the AGM, our results are incomparable to theirs.

Our paper analyzes SIM-EXT for Bulletproofs [16] and Spartan [54], two transparent zkSNARKs built from discrete-log assumptions. There is a line of related work building similar SNARKs, such as Hyrax [57], and extensions to recursive composition like Halo [15] and Nova [44]. We suspect our techniques would extend to these constructions, and leave extending them to future work.

A key technical tool our results rely on is a "tree-builder" for proving knowledge soundness of NIZKs built from multi-round interactive arguments. As described above, our approach is a generalization of a beautiful recent work by Attema et al. [2]. This work develops a tree-builder for *perfect* special sound protocols which are extractable given a tree of *distinct* verifier challenges; we generalize their result to support *computational* special soundness and to allow different conditions on verifier challenges. Wikstrom [59] gives an alternate construction and analysis of a tree-builder which could have served as a starting point for us; however, their extractor has a worse concrete running time and tightness than Attema et al. In a revision of [17], the authors generalize Attema et al.'s tree builder to handle general predicates on prover messages; since we need more general predicates on verifier challenges, their generalization is not directly useful to us. Other recent works [33,42] analyze the knowledge soundness of Bulletproofs in the AGM/GGM without using an explicit tree-builder by, for example, going through the notion of round-by-round soundness [10].

*Concurrent work.* After the acceptance of this paper, Ganesh et al. [31] updated their ePrint version to contain a proof that Bulletproofs satisfy SIM-EXT in the ROM, removing the need for the AGM as in their conference version [30]. We note that their technique is somewhat different from ours, and that our results

additionally include proving that Spartan satisfies SIM-EXT. We leave a more detailed comparison of our work with theirs to the full version.

## 2 Preliminaries

We use $\mathbb{F}$ to denote a finite field with $\mathbb{F}^* = \mathbb{F} - \{0\}$, and $\lambda$ to denote the security parameter. For $k, n \in \mathbb{N}$, we denote $[k, n] = \{k, k+1, \ldots, n\}$, and $[n] = [1, n]$. We denote uniform sampling from a set $S$ by $a \xleftarrow{\$} S$. We denote vectors by boldface, e.g. $\mathbf{g} = (g_1, \ldots, g_n)$, and write $\mathbf{g}^{\mathbf{a}}$ to mean $g_1^{a_1} \cdot \cdots \cdot g_n^{a_n}$. We denote the length of a vector $\mathbf{a}$ by $|\mathbf{a}|$, the inner product between two vectors $\mathbf{a}, \mathbf{b}$ by $\mathbf{a} \cdot \mathbf{b}$ or $\langle \mathbf{a}, \mathbf{b} \rangle$, the Hadamard (entry-wise) product by $\mathbf{a} \circ \mathbf{b}$, and the tensor product by $\mathbf{a} \otimes \mathbf{b} = (a_1 b_1, \ldots, a_1 b_m, \ldots, a_n b_1, \ldots, a_n b_m)$.

Our relations are of the form $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ and are efficiently decidable, e.g. there exists a deterministic polynomial time algorithm that given $(\mathsf{pp}, x, w)$ outputs whether $(\mathsf{pp}, x, w) \in \mathcal{R}$. We abbreviate PPT for probabilistic polynomial time, and EPT for expected (probabilistic) polynomial time.

We use *code-based games* [6] to define many of our security notions. A game $\mathsf{G}_{\mathsf{S}}^{\mathcal{A}_1, \ldots, \mathcal{A}_n}$ denotes a run of parties $\mathcal{A}_1, \ldots, \mathcal{A}_n$ on a pre-specified set of procedures given by $\mathsf{S}$, returning a bit $b \in \{0,1\}$. We denote $\Pr[\mathsf{G}_{\mathsf{S}}^{\mathcal{A}_1, \ldots, \mathcal{A}_n}]$ the probability over the random coins used by $\mathsf{S}$ and all adversaries that the game's output is 1.

### 2.1 Assumptions

We assume the existence of a group generator generating *global public parameters* $\mathsf{pp}_{\mathcal{G}} := (\mathbb{G}, \mathbb{F}) \leftarrow \mathsf{GroupGen}(1^\lambda)$, where $\mathbb{G}$ is a group of prime order, with $\mathbb{F}$ as the corresponding field. These global parameters are used in the setup phase of every protocol we consider. We also assume a generator sampling procedure $g_1, \ldots, g_n \xleftarrow{\$} \mathsf{GenSamp}(\mathbb{G}, n)$. For space reasons, we omit definitions of the standard discrete log (DL) and DL relation assumptions, and refer to reader to [33].

### 2.2 Interactive Arguments

We define an interactive argument for relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$.

**Definition 2.1.** *An* interactive argument *for a relation $\mathcal{R}$ is a tuple of PPT algorithms $\Pi = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ with the following syntax:*

- $\mathsf{Setup}(\mathsf{pp}_{\mathcal{G}}) \to \mathsf{pp}$ : *outputs public parameters* $\mathsf{pp}$ *given global parameters* $\mathsf{pp}_{\mathcal{G}}$*,*
- $\langle \mathcal{P}(w), \mathcal{V} \rangle(\mathsf{pp}, x) \to \{0,1\}$ : *an interactive protocol whereby the prover $\mathcal{P}$, holding a witness $w$, interacts with the verifier $\mathcal{V}$ on common input $(\mathsf{pp}, x)$ to convince $\mathcal{V}$ that $(\mathsf{pp}, x, w) \in \mathcal{R}$. At the end, $\mathcal{V}$ outputs a bit for accept/reject.*

In the definition above, we assume the existence of a global setup algorithm $\mathsf{pp}_{\mathcal{G}} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$ (see Section 2.1), run once and for all before the setup phase of any interactive argument. For space reasons, we refer to reader to e.g. [17] for standard definitions of completeness, knowledge soundness, and honest-verifier zero knowledge for interactive arguments.

**Definition 2.2 (Public-Coin).** *An interactive argument* $\Pi = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ *is* public-coin *if in each round* $i$ *the verifier* $\mathcal{V}$ *samples its message uniformly at random from some challenge space* $\mathsf{Ch}_i$*, and uses no other randomness.*

Any public-coin interactive argument has a general $(2r+2)$-message, or equivalently, $(r+1)$-round format where the verifier sends the 0-th message, and the prover sends the last message. In particular, the transcript is of the form $\mathsf{tr} = (c_0, a_1, c_1, \ldots, a_r, c_r, a_{r+1})$, where $(a_1, \ldots, a_{r+1})$ are the prover's messages and $(c_0, \ldots, c_r)$ are the verifier's messages. Additionally, we have $c_0 = \emptyset$ in all protocols we consider, so that we will only consider $(2r+1)$-message protocols (where the prover sends the first and last message).

### 2.3 Non-Interactive Arguments in the ROM

In practice, we often use the *Fiat-Shamir transform* (see Section 2.4) to compile public-coin interactive arguments into their *non-interactive* versions, in a model where both parties have black-box access to a *random oracle*, i.e. a uniformly sampled function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\lambda$. For public-coin $(2r+1)$-message interactive arguments with challenge spaces $\mathsf{Ch}_1, \ldots, \mathsf{Ch}_r$, we will actually need $r$ independent random oracles $\mathsf{H}_i : \{0,1\}^* \to \mathsf{Ch}_i$ with $i \in [1, r]$. For simplicity, we will denote these by a single random oracle $\mathsf{H}$, and it will be clear from context which random oracle is being used in a given round.

**Definition 2.3.** *A* non-interactive argument (NARG) *in the ROM for a relation* $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ *is a tuple of algorithms* $\Pi = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$*, with* $\mathcal{P}, \mathcal{V}$ *having black-box access to a random oracle* $\mathsf{H}$*, with the following syntax:*

- $\mathsf{Setup}(\mathsf{pp}_{\mathcal{G}}) \to \mathsf{pp}$ *generates the public parameters,*
- $\mathcal{P}^{\mathsf{H}}(\mathsf{pp}, x, w) \to \pi$ *generates a proof given* $\mathsf{pp}$ *and an input-witness pair* $(x, w)$*,*
- $\mathcal{V}^{\mathsf{H}}(\mathsf{pp}, x, \pi) \to \{0,1\}$ *checks if proof* $\pi$ *is valid for* $\mathsf{pp}$ *and input* $x$*.*

We define the following properties of NARGs:

- **Completeness.** For every adversary $\mathcal{A}$,
$$\Pr\left[\begin{matrix}(\mathsf{pp}, x, w) \notin \mathcal{R} \vee \\ \mathcal{V}^{\mathsf{H}}(\mathsf{pp}, x, \pi) = 1\end{matrix} \;:\; \begin{matrix}\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}}) \\ (x, w) \leftarrow \mathcal{A}^{\mathsf{H}}(\mathsf{pp}) \\ \pi \leftarrow \mathcal{P}^{\mathsf{H}}(\mathsf{pp}, x, w)\end{matrix}\right] = 1.$$
- **Knowledge Soundness.** $\Pi$ is *(adaptively) knowledge sound (*KS*)* if there exists an extractor $\mathcal{E}$ running in expected polynomial time such that for every PPT adversary $\mathcal{P}^*$, the following probability is negligible in $\lambda$:
$$\mathbf{Adv}^{\mathsf{KS}}_{\Pi_{\mathsf{FS}}, \mathcal{R}}(\mathcal{E}, \mathcal{P}^*) := \left|\Pr[\mathsf{KS}^{\mathcal{P}^*}_{0, \Pi_{\mathsf{FS}}}(\lambda)] - \Pr[\mathsf{KS}^{\mathcal{E}, \mathcal{P}^*}_{1, \Pi_{\mathsf{FS}}, \mathcal{R}}(\lambda)]\right|.$$
The knowledge soundness games are defined in Figure 1.

We define zero-knowledge in a model where the random oracle is *explicitly-programmable* [58] by the simulator. Here, the simulator $\mathcal{S}$ can reprogram the random oracle $\mathsf{H}$, and this modified oracle is provided to the distinguisher.

| **Game** $\mathsf{KS}^{\mathcal{P}^*}_{0,\Pi_{\mathsf{FS}}}(\lambda)$ | **Game** $\mathsf{KS}^{\mathcal{E},\mathcal{P}^*}_{1,\Pi_{\mathsf{FS}},\mathcal{R}}(\lambda)$ |
|---|---|
| $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ | $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ |
| $(x, \pi) \leftarrow (\mathcal{P}^*)^{\mathsf{H}}(\mathsf{pp})$ | $(x, \pi) \leftarrow (\mathcal{P}^*)^{\mathsf{H}}(\mathsf{pp})$ |
| $b \leftarrow \mathcal{V}^{\mathsf{H}}_{\mathsf{FS}}(\mathsf{pp}, x, \pi)$ | $b \leftarrow \mathcal{V}^{\mathsf{H}}_{\mathsf{FS}}(\mathsf{pp}, x, \pi)$ |
| **return** $b$ | $w \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp}, x, \pi)$ |
| | **return** $b \wedge (\mathsf{pp}, x, w) \in \mathcal{R}$ |

Fig. 1: Knowledge soundness security games. Here the extractor $\mathcal{E}$ is given black-box access to $\mathcal{P}^*$. In particular, $\mathcal{E}$ implements $\mathsf{H}$ for $\mathcal{P}^*$ and can rewind $\mathcal{P}^*$ to any point.

| **Game** $\mathsf{ZK}^{\mathcal{D},\mathcal{P}}_{0,\Pi_{\mathsf{FS}},\mathcal{R}}(\lambda)$ | **Game** $\mathsf{ZK}^{\mathcal{D},\mathcal{S}}_{1,\Pi_{\mathsf{FS}},\mathcal{R}}(\lambda)$ |
|---|---|
| $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ | $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ |
| $b \leftarrow \mathcal{D}^{\mathsf{H}(\cdot),\mathcal{P}'(\mathsf{pp},\cdot,\cdot)}(1^\lambda)$ | $b \leftarrow \mathcal{D}^{\mathsf{H}(\cdot),\mathcal{S}'(\mathsf{pp},\cdot,\cdot)}(1^\lambda)$ |
| **return** $b$ | **return** $b$ |
| $\mathcal{P}'(\mathsf{pp}, x, w)$ | $\mathcal{S}'(\mathsf{pp}, x, w)$ |
| **if** $(\mathsf{pp}, x, w) \notin \mathcal{R}$ **then return** $\perp$ | **if** $(\mathsf{pp}, x, w) \notin \mathcal{R}$ **then return** $\perp$ |
| **else return** $\mathcal{P}(\mathsf{pp}, x, w)$ | **else return** $\mathcal{S}^{\mathsf{RePro}}(\mathsf{pp}, x)$ |

Fig. 2: Zero-knowledge security games. Here the simulator $\mathcal{S}$ gets access to a $\mathsf{RePro}$ oracle that on input $(a, b)$ reprograms $\mathsf{H}(a) := b$.

**Definition 2.4 (Zero-Knowledge).** *$\Pi$ satisfies* (statistical) unbounded non-interactive zero-knowledge *(NIZK) if there exists a PPT simulator $\mathcal{S}$ such that for $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ and any unbounded distinguisher $\mathcal{D}$, the following probability is negligible in $\lambda$:*

$$\mathbf{Adv}^{ZK}_{\Pi_{\mathsf{FS}},\mathcal{R}}(\mathcal{S}, \mathcal{D}) := \left| \Pr\left[ ZK^{\mathcal{D},\mathcal{P}}_{0,\Pi_{\mathsf{FS}},\mathcal{R}}(\lambda) \right] - \Pr\left[ ZK^{\mathcal{D},\mathcal{S}}_{1,\Pi_{\mathsf{FS}},\mathcal{R}}(\lambda) \right] \right|.$$

*The zero-knowledge games are defined in Figure 2.*

### 2.4 The Fiat-Shamir Transformation

We define the Fiat-Shamir transform [25], which removes interaction from any public-coin interactive argument.

**Definition 2.5 (Fiat-Shamir Transformation).** *Let $\Pi = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ be a public-coin $(2r+1)$-message interactive argument of knowledge. Denote the transcript as $\mathsf{tr} = (a_1, c_1, \ldots, a_r, c_r, a_{r+1})$. The* Fiat-Shamir transformation *turns $\Pi$ into a non-interactive protocol $\Pi_{\mathsf{FS}}$ in the ROM, where:*

- $\mathsf{Setup}_{\mathsf{FS}}(\mathsf{pp}_{\mathcal{G}})$ *is the same as* $\mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$*,*

- *the prover* $\mathcal{P}_{\mathsf{FS}}$*, on input* $(\mathsf{pp}, x, w)$*, invokes* $\mathcal{P}(x, w)$*, and instead of asking the verifier for challenge* $c_i$ *in round* $i$*, queries the random oracle to get*

$$c_i = \mathsf{H}(\mathsf{pp}, x, a_1, \ldots, a_i) \text{ for all } i = 1, \ldots, r.$$

  $\mathcal{P}_{\mathsf{FS}}$ *then outputs a non-interactive proof* $\pi = (a_1, \ldots, a_r, a_{r+1})$*.*
- *the verifier* $\mathcal{V}_{\mathsf{FS}}$*, on input* $(\mathsf{pp}, x, \pi)$*, derives challenges* $c_i$*'s by querying the random oracle as* $\mathcal{P}_{\mathsf{FS}}$ *does, then runs* $\mathcal{V}(\mathsf{pp}, x, (a_1, c_1, \ldots, a_r, c_r, a_{r+1}))$ *and outputs what* $\mathcal{V}$ *outputs.*

For all protocols $\Pi$ considered in this paper, it is clear that both $\Pi$ and $\Pi_{\mathsf{FS}}$ satisfy (perfect) completeness. Furthermore, $\Pi_{\mathsf{FS}}$ satisfies knowledge soundness if $\Pi$ is *(computationally) special sound* (see Section 4). For zero-knowledge, we have a *canonical simulator* $\mathcal{S}_{\mathsf{FS}}$ for $\Pi_{\mathsf{FS}}$ based on any HVZK simulator $\mathcal{S}$ for $\Pi$.

**Definition 2.6 (Canonical Simulator).** *Let* $\Pi$ *be a public-coin interactive argument with HVZK simulator* $\mathcal{S}$*. Define the* canonical simulator $\mathcal{S}_{\mathsf{FS}}$ *for* $\Pi_{\mathsf{FS}}$ *to be an algorithm that on input* $(\mathsf{pp}, x)$ *runs* $\mathcal{S}(\mathsf{pp}, x)$ *to get a transcript* $\mathsf{tr} = (a_1, c_1, \ldots, a_r, c_r, a_{r+1})$*, then reprogram* $\mathsf{H}(\mathsf{pp}, x, a_1, \ldots, a_i) := c_i$ *for all* $i \in [r]$*.*

*Remark 2.7.* It can be shown that $\mathcal{S}_{\mathsf{FS}}$ is a NIZK simulator for $\Pi_{\mathsf{FS}}$ if $\mathcal{S}$ is an HVZK simulator and the fact that the first message $a_1$ has sufficient min-entropy [24,30]. Looking ahead, given *any* simulator $\mathcal{S}$ for $\Pi_{\mathsf{FS}}$, to show that it is a NIZK simulator, it suffices to show that $\mathcal{S}$ produces indistinguishable transcripts $\mathsf{tr} = (a_1, c_1, \ldots, a_{r+1})$ from honestly generated transcripts, and that the first message $a_1$ has sufficient min-entropy.

## 3 Simulation Extractability

We define the central notion of our work, simulation extractability (SIM-EXT), which requires that extractability holds even when the malicious prover is given access to simulated proofs. SIM-EXT implies adaptive (knowledge) soundness and non-malleability for the proof system [30,43,50], and allows building secure *signatures of knowledge* via standard transforms [19,39].

**Definition 3.1 (Simulation Extractability).** *Let* $\Pi = (\mathsf{Setup}, \mathcal{P}, \mathcal{V})$ *be a public-coin zero-knowledge interactive argument for relation* $\mathcal{R}$ *with associated NIZK* $\Pi_{\mathsf{FS}} = (\mathsf{Setup}, \mathcal{P}_{\mathsf{FS}}, \mathcal{V}_{\mathsf{FS}})$*. We say* $\Pi_{\mathsf{FS}}$ *satisfies* simulation extractability (SIM-EXT) *with respect to a simulator* $\mathcal{S}$ *if there exists an efficient simulator-extractor* $\mathcal{E}$ *such that for every PPT adversary* $\mathcal{P}^*$*, the following probability is negligible in* $\lambda$*:*

$$\mathbf{Adv}^{\mathsf{SIM\text{-}EXT}}_{\Pi_{\mathsf{FS}}, \mathcal{R}}(\mathcal{S}, \mathcal{E}, \mathcal{P}^*, \lambda) := \left| \Pr[\mathsf{SIM\text{-}EXT}^{\mathcal{S}, \mathcal{P}^*}_{0, \Pi_{\mathsf{FS}}}(\lambda)] - \Pr[\mathsf{SIM\text{-}EXT}^{\mathcal{E}, \mathcal{S}, \mathcal{P}^*}_{1, \Pi_{\mathsf{FS}}, \mathcal{R}}(\lambda)] \right|.$$

*Games* $\mathsf{SIM\text{-}EXT}_0$ *and* $\mathsf{SIM\text{-}EXT}_1$ *are defined in Figure 3.*

| **Game** $\mathsf{SIM\text{-}EXT}^{\mathcal{S},\mathcal{P}^*}_{0,\Pi_{\mathsf{FS}}}(\lambda)$ | **Game** $\mathsf{SIM\text{-}EXT}^{\mathcal{E},\mathcal{S},\mathcal{P}^*}_{1,\Pi_{\mathsf{FS}},\mathcal{R}}(\lambda)$ |
|---|---|
| $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ | $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ |
| $(x,\pi) \leftarrow (\mathcal{P}^*)^{\mathsf{H},\mathcal{S}}(\mathsf{pp})$ | $(x,\pi) \leftarrow (\mathcal{P}^*)^{\mathsf{H},\mathcal{S}}(\mathsf{pp})$ |
| $b \leftarrow \mathcal{V}^{\mathsf{H}'}_{\mathsf{FS}}(\mathsf{pp},x,\pi)$ | $b \leftarrow \mathcal{V}^{\mathsf{H}'}_{\mathsf{FS}}(\mathsf{pp},x,\pi)$ |
| **return** $b \wedge (x,\pi) \notin \mathcal{Q}_{\mathsf{Sim}}$ | $w \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp},x,\pi)$ |
| | **return** $b \wedge (x,\pi) \notin \mathcal{Q}_{\mathsf{Sim}} \wedge (\mathsf{pp},x,w) \in \mathcal{R}$ |

Fig. 3: SIM-EXT security games. In both games, $\mathcal{S}$ returns a proof $\pi$ upon an input $x$ (and may reprogram the random oracle), while $\mathcal{Q}_{\mathsf{Sim}}$ records all pairs $(x,\pi)$ queried by $\mathcal{P}^*$. $\mathsf{H}'$ denotes the modified RO after all proof simulation queries. $\mathcal{E}$ is given black-box access to $\mathcal{P}^*$; in particular, it implements $\mathsf{H}$ and $\mathcal{S}$ for $\mathcal{P}^*$ and can rewind $\mathcal{P}^*$ to any point in its execution (with same initial randomness).

We will state an adaptation of the results in [29], which establishes a general theorem about simulation extractability. In particular, the authors of [29] define the notion of a *k-zero-knowledge* simulator that only needs to reprogram the random oracle in round $k$. Similarly, they define a property of *k-unique response*, which roughly states that the malicious prover's responses are uniquely determined after round $k$. Together, these two properties (for the same $k$) along with knowledge soundness will be enough to show simulation extractability.

**Definition 3.2 ($k$-Zero-Knowledge).** *Let $\Pi = (\mathsf{Setup},\mathcal{P},\mathcal{V})$ be a $(2r+1)$-message public-coin interactive argument with HVZK simulator $\mathcal{S}$, and $k \in [1,r]$. Let $\Pi_{\mathsf{FS}}$ be its associated FS-transformed NIZK. We say $\Pi_{\mathsf{FS}}$ satisfies* (perfect) $k$-zero-knowledge ($k$-ZK) *if there exists a zero-knowledge simulator $\mathcal{S}_{\mathsf{FS},k}$ that only needs to program the random oracle in round $k$, and whose output is identically distributed to that of honestly generated proofs.*

**Definition 3.3 ($k$-Unique Response).** *Let $\Pi = (\mathsf{Setup},\mathcal{P},\mathcal{V})$ be a $(2r+1)$-message public-coin interactive argument, with $\Pi_{\mathsf{FS}}$ its associated FS-transformed* NARG *and $k \in [0,r]$. We say $\Pi_{\mathsf{FS}}$ satisfies* $k$-unique response ($k$-UR) *if for all PPT adversaries $\mathcal{A}$, the following probability (defined with respect to the game in Figure 4) is negligible in $\lambda$:*

$$\mathbf{Adv}^{k\text{-}\mathsf{UR}}_{\Pi_{\mathsf{FS}}}(\mathcal{A}) := \Pr\left[k\text{-}\mathsf{UR}^{\mathcal{A}}_{\Pi_{\mathsf{FS}}}(\lambda)\right].$$

*When $k = 0$, we say that $\Pi_{\mathsf{FS}}$ has* (computationally) unique proofs.

We now state a key theorem that relates SIM-EXT to these properties; it is similar to the SIM-EXT theorem given in [29], with SRS update oracles removed. We give the proof in the full version.

**Theorem 3.4.** *Let $\Pi_{\mathsf{FS}}$ be a Fiat-Shamir compiled non-interactive argument for relation $\mathcal{R}$ from a $(2r+1)$-message public-coin interactive argument $\Pi$. Assume $\Pi_{\mathsf{FS}}$ satisfies* KS*, has a perfect $k$-ZK simulator $\mathcal{S}_{\mathsf{FS},k}$ for $k \in [1,r]$, and satisfies $k$-UR (for the same $k$). Then $\Pi_{\mathsf{FS}}$ satisfies* SIM-EXT*.*

$$\begin{array}{|l|}\hline \textbf{Game } k\text{-}\mathsf{UR}^{\mathcal{A}}_{\Pi_{\mathsf{FS}}}(\lambda) \\ \hline \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{pp}_{\mathcal{G}}) \\ (x, \pi, \pi', c) \leftarrow \mathcal{A}^{\mathsf{H}}(\mathsf{pp}) \\ b \leftarrow \mathcal{V}^{\mathsf{H}[(\mathsf{pp},x,\pi|_k)\mapsto c]}_{\mathsf{FS}}(\mathsf{pp}, x, \pi) = 1 \\ b' \leftarrow \mathcal{V}^{\mathsf{H}[(\mathsf{pp},x,\pi'|_k)\mapsto c]}_{\mathsf{FS}}(\mathsf{pp}, x, \pi') = 1 \\ \textbf{return } b \ \wedge \ b' \ \wedge \ \pi \neq \pi' \ \wedge \ \pi|_k = \pi'|_k \\ \hline \end{array}$$

Fig. 4: Security game for $k$-unique response. Here $\mathsf{H}[(\mathsf{pp}, x, \pi|_k) \mapsto c]$ denotes the random oracle where the input $(\mathsf{pp}, x, \pi|_k)$ is reprogrammed to output $c$.

*Concretely, let $\mathcal{E}$ be a* KS *extractor for $\Pi_{\mathsf{FS}}$. There exists a* SIM-EXT *simulator-extractor $\mathcal{E}_{\mathsf{SE}}$ for $\Pi_{\mathsf{FS}}$ such that for every PPT prover $\mathcal{P}^*$ against $\Pi_{\mathsf{FS}}$ that makes at most $q_{\mathsf{H}}$ random oracle queries and $q_{\mathsf{Sim}}$ simulation queries, there exists another PPT prover $\mathcal{P}^*_{\mathsf{KS}}$ against* KS *and PPT adversary $\mathcal{A}$ against $k$-*UR *such that $\mathbf{Adv}^{\mathsf{SIM\text{-}EXT}}_{\Pi_{\mathsf{FS}},\mathcal{R}}(\mathcal{S}_{\mathsf{FS},k}, \mathcal{E}_{\mathsf{SE}}, \mathcal{P}^*) \leq \mathbf{Adv}^{\mathsf{KS}}_{\Pi_{\mathsf{FS}},\mathcal{R}}(\mathcal{E}, \mathcal{P}^*_{\mathsf{KS}}) + \mathbf{Adv}^{k\text{-}\mathsf{UR}}_{\Pi_{\mathsf{FS}}}(\mathcal{A}) + 2/|\mathsf{Ch}_k|$. Here $\mathsf{Ch}_k$ is the challenge set in round $k$. Furthermore, both $\mathcal{P}^*_{\mathsf{KS}}$ and $\mathcal{A}$ make at most $q_{\mathsf{H}}$ random oracle queries; their runtime is roughly equal to $\mathcal{P}^*$'s runtime plus $q_{\mathsf{Sim}}$ invocations of $\mathcal{S}_{\mathsf{FS},k}$. $\mathcal{E}_{\mathsf{SE}}$ is nearly as efficient as $\mathcal{E}$.*

## 4 Tree of Transcripts and Special Soundness

In this section, we show how to establish knowledge soundness (KS) of a FS-transformed protocol $\Pi_{\mathsf{FS}}$ based on the *computational special soundness* of the interactive protocol $\Pi$. The key is to construct an efficient *tree builder* $\mathcal{TB}$ that, given oracle access to a malicious prover $\mathcal{P}^*$ for $\Pi_{\mathsf{FS}}$, outputs a suitable *tree of accepting transcripts*, upon which a valid witness can be extracted.

**Definition 4.1 (Tree of Transcripts).** *Let $\Pi$ be a $(2r+1)$-message public-coin interactive argument for a relation $\mathcal{R}$, with challenge spaces $\mathsf{Ch}_1, \ldots, \mathsf{Ch}_r$. Given $\mathbf{n} = (n_1, \ldots, n_r) \in \mathbb{N}^r$ and $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_r)$ with $\phi_i : \mathsf{Ch}_i^{n_i} \to \{0,1\}$ for $i \in [r]$, we say that $\mathscr{T}$ is a $(\boldsymbol{\phi}, \mathbf{n})$-*tree of accepting transcripts *for* $\mathsf{pp}$ *if:*

1. *$\mathscr{T}$ is a tree of depth $r+1$,*
2. *For each $i \in [r+1]$, each vertex at depth $i$ is labeled with a prover's $i$-th message $a_i$, and if $i \leq r$, has exactly $n_i$ outgoing edges to its children, with each edge labeled with a verifier's $i$-th challenge $c_{i,1}, \ldots, c_{i,n_i}$ satisfying $\phi_i(c_{i,1}, \ldots, c_{i,n_i}) = 1$. Additionally, the root's label is prepended with $x$ (so the label becomes $(x, a_1)$),*
3. *The labels on any root-to-leaf path form a valid input-transcript pair $(x, \mathsf{tr})$.*

*We additionally define $\mathscr{T}$ to be accepting with respect to a input-transcript pair $(x, \mathsf{tr})$ if $(x, \mathsf{tr})$ corresponds to the left-most path of $\mathscr{T}$. We define a predicate $\mathsf{IsAccepting}((\boldsymbol{\phi}, \mathbf{n}), \mathsf{pp}, x, (\pi, )\mathscr{T})$ to check whether $\mathscr{T}$ is a $(\boldsymbol{\phi}, \mathbf{n})$-tree of accepting transcripts for $\mathsf{pp}$ and $x$, and optionally $\pi$.*

The usual definition of a tree of accepting transcripts [2,14] has $\phi_i$ be the predicate that the $i$-th challenges $c_{i,1}, \ldots, c_{i,n_i}$, coming from a vertex at depth $i$, are distinct (we call this the *distinctness predicate*). In that case, we will also abbreviate $\mathscr{T}$ as a $\mathbf{n}$-tree of accepting transcripts. However, we will need to consider more general *partition predicates* in our proofs of knowledge soundness for Spartan and Bulletproofs.

**Definition 4.2 (Partition Predicate).** *Let* $\mathsf{Ch} = \mathsf{Ch}^{(1)} \sqcup \mathsf{Ch}^{(2)} \cdots \sqcup \mathsf{Ch}^{(C)}$ *be a partition* $\mathscr{P}$ *of a set* $\mathsf{Ch}$ *into* $C$ *blocks. We assume the partition is* efficient*, i.e. given an index* $i \in [C]$*, we can enumerate the set* $\mathsf{Ch}^{(i)}$ *in polynomial time. For* $n \in \mathbb{N}$*, we define the corresponding* partition predicate $\phi_{\mathscr{P},n} : \mathsf{Ch}^n \to \{0,1\}$ *to be* $\phi_{\mathscr{P},n}(c_1, \ldots, c_n) = 1$ *if and only if* $c_1, \ldots, c_n$ *belong in distinct blocks of* $\mathsf{Ch}$*.*

*Remark 4.3.* Looking ahead, we will consider the following partition predicates:

- When $\mathsf{Ch} = \mathbb{F}^*$ is partitioned into $\{x, -x\}$ for all $x$. We abbreviate this predicate into the number $n$ of challenges as $n_{\pm}$.
- When $\mathsf{Ch} = \mathbb{F}^2$ is partitioned into $\{c \cdot x \mid c \in \mathbb{F}^*\}$ for all $x \in \{(0,0), (0,1)\} \cup \{(1,a) \mid a \in \mathbb{F}\}$ (this implies linear independence between two vectors). We abbreviate this predicate into the number $n$ of challenges as $n_{\mathsf{li}}$.

We now state a theorem asserting the existence of an efficient tree-builder that can generate $(\boldsymbol{\phi}, \mathbf{n})$-trees of accepting transcripts, where $\boldsymbol{\phi}$ consists of partition predicates as defined above. In the full version, we give a proof of this theorem along with a comparison of our tree-builder with that of Wikström [59].

**Theorem 4.4 (Efficient Tree Builder).** *Let* $\Pi$ *be a* $(2r+1)$*-message public-coin interactive argument with challenge spaces* $\mathsf{Ch}_1, \ldots, \mathsf{Ch}_r$*. Consider any efficiently decidable partition* $\mathsf{Ch}_i = \sqcup_{j=1}^{C_i} \mathsf{Ch}_{i,j}$ *with minimum partition size* $C = \min_i C_i$*, and let* $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_r)$ *be the corresponding partition predicate. Consider any* $\mathbf{n} = (n_1, \ldots, n_r) \in \mathbb{N}^r$ *with* $N = \prod_{i=1}^r n_i$*.*

*There exists a probabilistic algorithm* $\mathcal{TB}$ *for* $\Pi_{\mathsf{FS}}$ *with the following guarantees: given oracle access to a malicious prover* $\mathcal{P}^*$ *for* $\Pi_{\mathsf{FS}}$ *with success probability* $\epsilon(\mathcal{P}^*) := \Pr[\mathsf{KS}^{\mathcal{P}^*}_{0,\Pi_{\mathsf{FS}}}]$*,* $\mathcal{TB}$ *wins the tree-building game* $\mathsf{TreeBuild}^{\mathcal{TB},\mathcal{P}^*}_{\Pi_{\mathsf{FS}},(\boldsymbol{\phi},\mathbf{n})}$ *(shown in Figure 5) with probability at least*

$$\Pr\left[\mathsf{TreeBuild}^{\mathcal{TB},\mathcal{P}^*}_{\Pi_{\mathsf{FS}},(\boldsymbol{\phi},\mathbf{n})}\right] \geq \epsilon(\mathcal{P}^*) - \frac{Q(Q-1)/2 + (Q+1)\left(\sum_{i=1}^r n_i - r\right)}{C}.$$

*Furthermore,* $\mathcal{TB}$ *makes in expectation at most* $(Q+1)(N-1)+1$ *rewinding calls to* $\mathcal{P}^*$*, where* $Q$ *is an upper bound on the number of RO queries of* $\mathcal{P}^*$*.*

We now define computational special soundness, which stipulates the existence of a tree-extraction procedure $\mathcal{TE}$ that, given an appropriate tree of accepting transcripts produced by an efficient adversary, outputs a witness with high probability.

| **Game** $\mathsf{TreeBuild}^{\mathcal{TB},\mathcal{P}^*}_{\Pi_{\mathsf{FS}},(\boldsymbol{\phi},\mathbf{n})}(\lambda)$ | **Game** $\mathsf{SS}^{\mathcal{TE},\mathcal{A}}_{\Pi,\mathcal{R},(\boldsymbol{\phi},\mathbf{n})}(\lambda)$ |
|---|---|
| $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ | $\mathsf{pp} \leftarrow \mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$ |
| $(x, \pi) \leftarrow (\mathcal{P}^*)^{\mathsf{H}}(\mathsf{pp})$ | $(x, \mathscr{T}) \leftarrow \mathcal{A}(\mathsf{pp})$ |
| $\mathscr{T} \leftarrow \mathcal{TB}^{\mathcal{P}^*}(\mathsf{pp}, x, \pi)$ | $w \leftarrow \mathcal{TE}(\mathsf{pp}, x, \mathscr{T})$ |
| **return** $(\mathcal{V}^{\mathsf{H}}(\mathsf{pp}, x, \pi) = 1) \wedge$ | **return** $(\mathsf{pp}, x, w) \notin \mathcal{R} \wedge$ |
| $\mathsf{IsAccepting}((\boldsymbol{\phi}, \mathbf{n}), \mathsf{pp}, x, \pi, \mathscr{T})$ | $\mathsf{IsAccepting}((\boldsymbol{\phi}, \mathbf{n}), \mathsf{pp}, x, \mathscr{T})$ |

Fig. 5: Games for tree-building and special soundness. Here the tree-builder $\mathcal{TB}$ is given black-box access to $\mathcal{P}^*$. In particular, $\mathcal{TB}$ implements $\mathsf{H}$ for $\mathcal{P}^*$ and can rewind $\mathcal{P}^*$ to any point in its execution.

**Definition 4.5 (Special Soundness).** *Let $\Pi$ be a $(2r+1)$-message public-coin interactive argument for a relation $\mathcal{R}$ with challenge spaces $\mathsf{Ch}_1, \ldots, \mathsf{Ch}_r$. For any $\mathbf{n} = (n_1, \ldots, n_r) \in \mathbb{N}^r$ and $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_r)$ with $\phi_i : \mathsf{Ch}_i^{n_i} \to \{0,1\}$, we say $\Pi$ is* $(\boldsymbol{\phi}, \mathbf{n})$-computational special sound *if there exists a PPT tree-extraction algorithm $\mathcal{TE}$ such that for all EPT adversary $\mathcal{A}$, the following probability is negligible in $\lambda$:*

$$\mathbf{Adv}^{\mathsf{SS}}_{\Pi,\mathcal{R},(\boldsymbol{\phi},\mathbf{n})}(\mathcal{TE}, \mathcal{A}) := \Pr\left[\mathsf{SS}^{\mathcal{TE},\mathcal{A}}_{\Pi,\mathcal{R},(\boldsymbol{\phi},\mathbf{n})}(\lambda)\right].$$

*The special soundness game is shown in Figure 5. We say $\Pi$ is* computational special sound ($\mathsf{SS}$) *if it is $(\boldsymbol{\phi}, \mathbf{n})$-computational special sound for some $\boldsymbol{\phi}$ and $\mathbf{n}$.*

Using Theorem 4.4 and Definition 4.5, we get the following consequence that computational special soundness for an interactive protocol implies knowledge soundness for its non-interactive version.

**Lemma 4.6.** *Let $\Pi$ be a $(2r+1)$-message public-coin interactive argument that is $(\boldsymbol{\phi}, \mathbf{n})$-computational special sound with tree extractor $\mathcal{TE}$, where $\mathbf{n} = (n_1, \ldots, n_r) \in \mathbb{N}^r$ and $\boldsymbol{\phi}$ is a partition predicate with minimum partition size $C$. Then $\Pi_{\mathsf{FS}}$ satisfies knowledge soundness. Concretely, there exists an EPT extractor $\mathcal{E}$ such that for every PPT adversary $\mathcal{P}^*$ against $\mathsf{KS}$ making at most $Q$ random oracle calls, there exists an EPT adversary $\mathcal{A}$ against $\mathsf{SS}$ such that*

$$\mathbf{Adv}^{\mathsf{KS}}_{\Pi_{\mathsf{FS}},\mathcal{R}}(\mathcal{E}, \mathcal{P}^*) \leq \frac{Q(Q-1)/2 + (Q+1)\left(\sum_{i=1}^{r} n_i - r\right)}{C} + \mathbf{Adv}^{\mathsf{SS}}_{\Pi,(\boldsymbol{\phi},\mathbf{n})}(\mathcal{TE}, \mathcal{A}).$$

*Both $\mathcal{E}$ and $\mathcal{A}$ runs in expected time that is at most $O(Q \cdot N)$ the runtime of $\mathcal{P}^*$.*

*Proof.* Our proof goes through a sequence of hybrids. $\mathsf{Hyb}_0$ is the game $\mathsf{KS}^{\mathcal{P}^*}_{0,\Pi_{\mathsf{FS}}}$. $\mathsf{Hyb}_1$ is the same as $\mathsf{Hyb}_0$, except we also run $\mathcal{TB}^{\mathcal{P}^*}(\mathsf{pp}, x, \pi) \to \mathscr{T}$ and output 0 if $\mathscr{T}$ is not a $(\boldsymbol{\phi}, \mathbf{n})$-tree of accepting transcripts with respect to $(\mathsf{pp}, x, \pi)$. Note that $\mathsf{Hyb}_1$ is the same as the game $\mathsf{TreeBuild}^{\mathcal{TB},\mathcal{P}^*}_{\Pi_{\mathsf{FS}},(\boldsymbol{\phi},\mathbf{n})}$. Using Theorem 4.4, we get

$$|\Pr[\mathsf{Hyb}_0] - \Pr[\mathsf{Hyb}_1]| \leq \frac{Q(Q-1)/2 + (Q+1)\left(\sum_{i=1}^{r} n_i - r\right)}{C}.$$

We define $\mathsf{Hyb}_2$ to be the same as $\mathsf{Hyb}_1$, except we also run $\mathcal{TE}(\mathsf{pp}, x, \mathscr{T}) \to w$ and output 0 if $(\mathsf{pp}, x, w) \notin \mathcal{R}$. We define the extractor $\mathcal{E}$ to be as follows: run $\mathcal{TB}^{\mathcal{P}^*}(\mathsf{pp}, x, \pi) \to \mathscr{T}$ to obtain a tree of accepting transcripts, then run $\mathcal{TE}(\mathsf{pp}, x, \mathscr{T}) \to w$ to obtain a witness. By definition of $\mathcal{E}$, we can see that $\mathsf{Hyb}_2$ is the same as the game $\mathsf{KS}^{\mathcal{E},\mathcal{P}^*}_{1,\Pi_{\mathsf{FS}},\mathcal{R}}$.

We now claim that there exists an adversary $\mathcal{A}$ against $\mathsf{SS}$ such that

$$|\Pr[\mathsf{Hyb}_1] - \Pr[\mathsf{Hyb}_2]| \leq \mathbf{Adv}^{\mathsf{SS}}_{\Pi,(\boldsymbol{\phi},\mathbf{n})}(\mathcal{TE}, \mathcal{A}).$$

We define $\mathcal{A}$ to be as follows: given oracle access to $\mathcal{P}^*$, $\mathcal{A}$ runs $(\mathcal{P}^*)^{\mathsf{H}}(\mathsf{pp}) \to (x, \pi)$ by simulating $\mathsf{H}$ for $\mathcal{P}^*$, then runs $\mathcal{TB}^{\mathcal{P}^*}(\mathsf{pp}, x, \pi) \to \mathscr{T}$, and outputs $(x, \mathscr{T})$. It is then straightforward to argue that $\mathsf{Hyb}_2$ returns 0 while $\mathsf{Hyb}_1$ returns 1 precisely when $\mathcal{A}$ wins in $\mathsf{SS}$. □

## 5 Simulation Extractability of Spartan

In this section, we use our general theorems to prove SIM-EXT of Spartan [54], a transparent zkSNARKs with security based on the discrete log assumption. [54] presents two version of Spartan, one with a linear verifier (called Spartan-NIZK) and one with a sublinear verifier (called Spartan-SNARK) achieved via encoding the R1CS matrices with a *sparse multilinear polynomial commitment*.

### 5.1 Spartan Protocols

We first describe the two variants of Spartan. Note that in a slight abuse of terminology, we will use Spartan-NIZK and Spartan-SNARK to refer to the interactive versions of their respective protocols. When we wish to refer specifically to the non-interactive versions, we will write $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ and $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$.

**Definition 5.1 (R1CS).** *A* R1CS instance *is a tuple* $(\mathbb{F}, A, B, C, m, n, \mathsf{io})$ *where* $A, B, C \in \mathbb{F}^{m \times m}$ *each with at most* $n = \Omega(m)$ *non-zero entries, and* $m \geq |\mathsf{io}| + 1$. *A* R1CS witness *is a vector* $w \in \mathbb{F}^{m-|\mathsf{io}|-1}$ *such that if* $Z = (\mathsf{io}, 1, w)$, *then* $(A \cdot Z) \circ (B \cdot Z) = C \cdot Z$.

Spartan *makes further assumptions on the* R1CS *instances, namely that* $m = 2^\mu, n = 2^\nu$ *are powers of two, and* $|\mathsf{io}| + 1 = |w| = m/2$.

*Key ideas.* Both the NIZK and SNARK variants of Spartan prove satisfiability of R1CS instances using roughly the same ideas we now outline. See Figure 6 for a protocol description. It uses the following sub-protocols (description in full version):

1. The Pedersen commitment scheme $\mathbf{g}^{\mathbf{a}} \cdot h^\omega \leftarrow \mathsf{Commit}((n, \mathbf{g}, h), \mathbf{a}; \omega)$.
2. Four $\Sigma$-protocols sharing the same setup:
   (a) OpenPf to prove knowledge of a commitment $C = g^x \cdot h^\omega$,
   (b) EqPf to prove equality of two commitments $C_1 = g^x \cdot h^{\omega_1}, C_2 = g^x \cdot h^{\omega_2}$,

(c) ProdPf to prove that three commitments $C_{v_1}, C_{v_2}, C_{v_3}$ satisfy $v_1 \cdot v_2 = v_3$,
(d) DotProdPf to prove that a multi-commitment $C_{\mathbf{x}}$ and a commitment $C_y$ satisfy $y = \langle \mathbf{x}, \mathbf{a} \rangle$ for a public vector $\mathbf{a}$,

3. A $(\mu + 1)$-round public-coin interactive protocol $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$ for proving polynomial evaluations of any multilinear polynomial $p(X_1, \ldots, X_\mu)$.
4. Additionally, in the case of Spartan-SNARK, we also need $\mathsf{PC}_{\mathsf{SparseMulti}}.\mathsf{Open}$ for proving evaluations of *sparse* multilinear polynomials $\widetilde{A}, \widetilde{B}, \widetilde{C}$.

At a high level, the main idea of Spartan is to reduce the satisfiability of the given R1CS instance to a claim that can be verified via sumcheck. To do this, the matrices $A, B, C$ are interpreted as functions $\{0,1\}^\mu \times \{0,1\}^\mu \to \mathbb{F}$, and similarly $Z : \{0,1\}^\mu \to \mathbb{F}$, by writing the indices as their binary representations. We then take the *multilinear extension* $\widetilde{A}, \widetilde{B}, \widetilde{C}, \widetilde{Z}$ of these functions, and define the polynomial

$$\widetilde{\mathcal{F}_{\mathsf{io}}}(X) = \left( \sum_{y \leftarrow \{0,1\}^\mu} \widetilde{A}(X, y) \cdot \widetilde{Z}(y) \right) \cdot \left( \sum_{y \leftarrow \{0,1\}^\mu} \widetilde{B}(X, y) \cdot \widetilde{Z}(y) \right) - \left( \sum_{y \leftarrow \{0,1\}^\mu} \widetilde{C}(X, y) \cdot \widetilde{Z}(y) \right).$$

Note that $\widetilde{\mathcal{F}_{\mathsf{io}}}(X)$ vanishes on $\{0,1\}^\mu$ if and only if the R1CS constraint is satisfied. Finally, we turn this vanishing condition into a sumcheck instance by defining $\mathcal{G}_{\mathsf{io},\tau}(X) = \widetilde{\mathcal{F}_{\mathsf{io}}}(X) \cdot \widetilde{\mathsf{eq}}(X, \tau)$ for a random $\tau \in \mathbb{F}^\mu$, supplied by the verifier. The goal is then to prove that $\sum_{y \in \{0,1\}^\mu} \mathcal{G}_{\mathsf{io},\tau}(y) = 0$. The prover and verifier engage in sumcheck for this claim. The final step of sumcheck requires the verifier to evaluate $\mathcal{G}_{\mathsf{io},\tau}$ at a random point $r_x$, but the verifier cannot do this itself; thus, the prover and verifier engage in another run of sumcheck (more precisely, three runs batched together with verifier randomness) to reduce the task of evaluating $\mathcal{G}_{\mathsf{io},\tau}(r_x)$ to evaluating $\widetilde{A}, \widetilde{B}, \widetilde{C}$ all at $(r_x, r_y)$, and $\widetilde{Z}$ at $r_y$. In both Spartan-NIZK and Spartan-SNARK, the verifier gets a commitment to the evaluation of the witness, and is convinced the committed evaluation is correct via $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$. (Our analyses below assume $\mathsf{PC}_{\mathsf{Multi}}$ is instantiated with HyraxPC [57].) In Spartan-NIZK, the verifier evaluates $\widetilde{A}, \widetilde{B}, \widetilde{C}$ itself; in Spartan-SNARK, the prover sends the verifier the evaluations and uses $\mathsf{PC}_{\mathsf{SparseMulti}}.\mathsf{Open}$, a secondary proof protocol, to convince the verifier of their correctness.

We can compute the number of rounds of Spartan-NIZK to be $\mathsf{r} = 7\mu + 11$. For Spartan-SNARK, the transcript is the same except for the verifier sending its commitments to $\widetilde{A}, \widetilde{B}, \widetilde{C}$ to the prover, the evaluations $v_1, v_2, v_3$, and the $\mathcal{O}(\mu)$-round transcript of $\mathsf{PC}_{\mathsf{SparseMulti}}.\mathsf{Open}$. Thus, the transcript of Spartan-SNARK has $O(\mu)$ more rounds for evaluating $\widetilde{A}(r_x, r_y), \widetilde{B}(r_x, r_y), \widetilde{C}(r_x, r_y)$.

### 5.2 SIM-EXT Analysis of Spartan-NIZK

Following Theorem 3.4, to prove that $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ satisfies SIM-EXT, we will need to show that it satisfies knowledge soundness (KS) along with $k$-ZK and $k$-UR for the same round $k$. By Lemma 4.6, knowledge soundness in turn depends on computational special soundness (SS) of the interactive protocol Spartan-NIZK. Our first set of results will be to establish SS of Spartan-NIZK

**R1CS Relation.**

$$\mathcal{R}_{\mathsf{R1CS}} = \left\{ \begin{array}{c} ((\mathbb{F}, m, n, A, B, C), \mathsf{io}, w) : \\ (A \cdot Z) \circ (B \cdot Z) = (C \cdot Z), \text{ where } Z = (\mathsf{io}, 1, w) \end{array} \right\}.$$

**Setup Phase.** Let $\mu = \log m$. Run $\mathsf{pp}_{\mathcal{G}} = (\mathbb{G}, \mathbb{F}) \leftarrow \mathsf{GroupGen}(1^{\lambda})$, $\mathsf{pp}_{\mathsf{Multi}} \leftarrow \mathsf{PC}_{\mathsf{Multi}}.\mathsf{Setup}(\mu, \mathsf{pp}_{\mathcal{G}})$ and $\mathsf{pp}_{\Sigma} \leftarrow \Sigma.\mathsf{Setup}(\mathsf{pp}_{\mathcal{G}})$. Run $\mathsf{pp}_{\mathsf{Sparse}} \leftarrow \mathsf{PC}_{\mathsf{Multi}}.\mathsf{Setup}(\mu, n, \mathsf{pp}_{\mathcal{G}})$. Return $\mathsf{pp} = (\mathsf{pp}_{\mathsf{Multi}}, \mathsf{pp}_{\Sigma}, \mathsf{pp}_{\mathsf{Sparse}})$.

**Interaction Phase.**

0. $\mathcal{V}$ computes $C_{\widetilde{X}} \leftarrow \mathsf{PC}_{\mathsf{SparseMulti}}.\mathsf{Commit}(\mathsf{pp}, \widetilde{X})$ for $X \in \{A, B, C\}$.

$\mathcal{V}$ then sends the coins used in this step to $\mathcal{P}$.

1. $\mathcal{P}$ computes $C_{\widetilde{w}} \leftarrow \mathsf{PC}_{\mathsf{Multi}}.\mathsf{Commit}(\mathsf{pp}_{\mathsf{PC}}, \widetilde{w})$ and sends $C_{\widetilde{w}}$ to $\mathcal{V}$.
2. $\mathcal{V}$ responds with challenge $\tau \xleftarrow{\$} \mathbb{F}^{\mu}$.
3. $\mathcal{P}, \mathcal{V}$ engage in sumcheck for $\sum_{x \in \{0,1\}^{\mu}} \mathcal{G}_{\mathsf{io},\tau}(x) \stackrel{?}{=} 0$.[a]

At the end, $\mathcal{P}$ sends $C_{e_x} \leftarrow \mathsf{Commit}(\mathsf{pp}, e_x)$ supposedly containing $e_x = \mathcal{G}_{\mathsf{io},\tau}(r_x)$ for $r_x \xleftarrow{\$} \mathbb{F}^{\mu}$ sent by $\mathcal{V}$.

4. $\mathcal{P}$ computes $v_M = \sum_{y \in \{0,1\}^{\mu}} \widetilde{M}(r_x, y) \cdot \widetilde{Z}(y)$ for $M \in \{A, B, C\}$. It then computes $C_{v_M} \leftarrow \mathsf{Commit}(\mathsf{pp}, v_M)$ for $M \in \{A, B, C\}$ and $C_{v_{AB}} \leftarrow \mathsf{Commit}(\mathsf{pp}, v_A \cdot v_B)$.

$\mathcal{P}$ sends $C_{v_A}, C_{v_B}, C_{v_C}, C_{v_{AB}}$ to $\mathcal{V}$.

5. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{ProdPf}$ to show that $v_{AB} = v_A \cdot v_B$.
6. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{OpenPf}$ to show that $C_{v_C}$ is indeed a commitment to $v_C$.
7. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{EqPf}$ to show that $e_x = (v_{AB} - v_C) \cdot \widetilde{\mathrm{eq}}(r_x, \tau)$.
8. $\mathcal{V}$ responds with challenges $r_A, r_B, r_C \xleftarrow{\$} \mathbb{F}$.
9. Let $\mathcal{H}_{r_x}(Y) = \sum_{M \in \{A,B,C\}} r_M \cdot \widetilde{M}(r_x, Y) \cdot \widetilde{Z}(Y)$ and $T = \sum_{M \in \{A,B,C\}} r_M \cdot v_M$. $\mathcal{P}, \mathcal{V}$ engage in sumcheck for $\sum_{y \in \{0,1\}^{\mu}} \mathcal{H}_{r_x}(y) = T$.

At the end, $\mathcal{P}$ sends a commitment $C_{e_y}$ supposedly containing $e_y = \mathcal{H}_{r_x}(r_y)$ for $r_y \xleftarrow{\$} \mathbb{F}^{\mu}$ sent by $\mathcal{V}$.

10. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$ for $\widetilde{w}((r_y)_{[1:]}) \to v_w$. At the end, both parties get $C_{v_w}$ and compute

$$C_{v_Z} = C_{v_w}^{1-(r_y)_0} \cdot C_{v_{\mathsf{io}}}^{(r_y)_0},$$

where $v_{\mathsf{io}} \leftarrow \widetilde{(\mathsf{io}, 1)}((r_y)_{[1:]})$ and $C_{v_{\mathsf{io}}} \leftarrow \mathsf{Commit}(\mathsf{pp}, v_{\mathsf{io}}; 0)$.

11. $\mathcal{V}$ computes $v_1 = \widetilde{A}(r_x, r_y), \quad v_2 = \widetilde{B}(r_x, r_y), \quad v_3 = \widetilde{C}(r_x, r_y)$.

Instead $\mathcal{V}$ receives $v_1, v_2, v_3$ from $\mathcal{P}$.

Then $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$ to check that $v_1, v_2, v_3$ are correct.

12. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{EqPf}$ to check that $e_y = (r_A \cdot v_1 + r_B \cdot v_2 + r_C \cdot v_3) \cdot v_Z$.

[a] The sumcheck subroutine is described in Figure 7.

Fig. 6: Spartan-NIZK, with modifications for Spartan-SNARK in red.

**Sumcheck Sub-Protocol.** The sumcheck relation is $\sum_{x\in\{0,1\}^\mu} p(x) = T$, where $p$ is a multivariate polynomial of individual degree at most $d$. $\mathcal{V}$ is given a commitment $C_p$ and a commitment $C_T$. The sumcheck subprotocol reduces this claim to the claim that $p(r_x) \stackrel{?}{=} e_x$ for a random $r_x \xleftarrow{\$} \mathbb{F}^\mu$ sampled randomly by $\mathcal{V}$, and some claimed value $e_x \in \mathbb{F}$ available as a commitment $C_{e_x}$ to $\mathcal{V}$.

Let $e_0 = T$. For $i = 1, \ldots, \mu$:

1. $\mathcal{P}$ computes the polynomial $p_i(X) = \sum_{x\in\{0,1\}^{\mu-i}} P(r_1, \ldots, r_{i-1}, X, x)$, parse it as a vector of coefficients, then sends $C_{p_i} \leftarrow \mathsf{Commit}(\mathsf{pp}, p_i; \omega_{p_i})$ to $\mathcal{V}$.
2. $\mathcal{V}$ responds with challenge $r_i \xleftarrow{\$} \mathbb{F}$.
3. $\mathcal{P}$ computes $e_i = p_i(r_i)$, then sends $C_{e_i} \leftarrow \mathsf{Commit}(\mathsf{pp}, e_i; \omega_{e_i})$ to $\mathcal{V}$.
4. $\mathcal{V}$ responds with challenges $w_{i,1}, w_{i,2} \xleftarrow{\$} \mathbb{F}$.
5. $\mathcal{P}, \mathcal{V}$ compute $\mathbf{a} = w_{i,1} \cdot (\mathbf{0}^k + \mathbf{1}^k) + w_{i,2} \cdot \mathbf{r_i}^k$ and $C_{y_i} = C_{e_{i-1}}^{w_{i,1}} \cdot C_{e_i}^{w_{i,2}}$. In addition, $\mathcal{P}$ computes $y_i = w_{i,1} \cdot e_{i-1} + w_{i,2} \cdot e_i$ and $\omega_{y_i} = w_{i,1} \cdot \omega_{e_{i-1}} + w_{i,2} \cdot \omega_{e_i}$.
6. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{DotProdPf}(\mathsf{pp}, (C_{p_i}, C_{y_i}, \mathbf{a}), (p_i, \omega_{p_i}, y_i, \omega_{y_i}))$.

Fig. 7: Sumcheck Sub-Protocol

through the following steps: (1) We first analyze the information-theoretic core of Spartan-NIZK, which is obtained from the protocol by sending all polynomials and evaluations in the clear, and checking the equalities directly. We call this variant Spartan-Core. (2) We then analyze how to extract from the various commitments and subprotocols in Spartan-NIZK to recover Spartan-Core.

The soundness of Spartan-Core has been analyzed in [54].

**Lemma 5.2 ([54]).** Spartan-Core *has soundness error* $\frac{6\mu+1}{|\mathbb{F}|}$.

Special soundness for $\Sigma$-protocols was analyzed in another previous work [57].

**Lemma 5.3 ([57]).** *Let* $\Pi \in \{\mathsf{OpenPf}, \mathsf{EqPf}, \mathsf{ProdPf}, \mathsf{DotProdPf}\}$. *Then* $\Pi$ *is* 2*-perfect special sound. Concretely, there exists a tree-extraction algorithm* $\mathcal{TE}_\Pi$ *that can extract a valid witness for* $\Pi$ *given any* 2*-tree of accepting transcripts.*

We also need to analyze special soundness of $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$. Note that while [57] introduced this protocol, they did not provide a concrete soundness result for it. The proof of the lemma below appears in the full version.

**Lemma 5.4.** $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$ *is* $\mathbf{n} = (\sqrt{m}, \underbrace{4_\pm, \ldots, 4_\pm}_{\mu/2}, 2)$*-computational special sound. Concretely, there exists a tree-extraction algorithm* $\mathcal{TE}_{\mathsf{PC}_{\mathsf{Multi}}}$ *such that for any EPT adversary* $\mathcal{A}$ *against* SS *of* $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$*, there exists an EPT adversary* $\mathcal{B}$ *against* DL-REL*, as efficient as* $\mathcal{A}$ *and* $\mathcal{TE}_{\mathsf{PC}_{\mathsf{Multi}}}$ *combined, such that*

$$\mathbf{Adv}^{\mathsf{SS}}_{\Pi,\mathbf{n}}(\mathcal{TE}_{\mathsf{PC}_{\mathsf{Multi}}}, \mathcal{A}) \leq \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G},\sqrt{m}+2}(\mathcal{B}).$$

Our next step is to analyze the computational special soundness of the sumcheck subprotocol in Figure 7. Since it is not strictly an interactive argument, we explicitly state the guarantees of the tree extractor.

**Lemma 5.5.** *There exists a tree extractor $\mathcal{TE}_{\mathsf{SC}}$ such that given a $(1, 2_{\mathsf{li}}, 2)^{\mu}$-tree of accepting transcripts, produced by an adversary $\mathcal{A}$, for the sumcheck subprotocol, either outputs polynomials $p_1(X), \ldots, p_{\mu}(X)$ that satisfy the information-theoretic sumcheck protocol, or we can build an adversary $\mathcal{B}$, as efficient as $\mathcal{TE}_{\mathsf{SC}}$ and $\mathcal{A}$ combined, against* DL-REL.

*Proof.* We will analyze a single iteration $i \in [\mu]$ of the sumcheck subprotocol; all other iterations will follow the same reasoning. We construct a tree extractor $\mathcal{TE}_{\mathsf{SC}}$ that does the following for each iteration $i \in [\mu]$: given a $(1, 2_{\mathsf{li}}, 2)$-tree of transcripts,

1. Run $\mathcal{TE}_{\mathsf{DotProdPf}}$ on each $(1, 1, 2)$-subtree to extract $(p_i, \omega_{p_i}, y_i, \omega_{y_i})$, where
$$C_{p_i} = \mathsf{PC.Commit}(\mathsf{pp}, p_i; \omega_{p_i}), \quad C_{y_i} = \mathsf{Commit}(\mathsf{pp}, y_i; \omega_{y_i}), \quad \langle p_i, \mathbf{a}_i \rangle = y_i,$$
and $y_i$ is supposedly equal to $w_{i,1} \cdot e_{i-1} + w_{i,2} \cdot e_i$.
2. Given two pairs of linearly independent challenges $(w_{i,1}, w_{i,2}), (w'_{i,1}, w'_{i,2})$, with extracted witnesses $(p_i, \omega_{p_i}, y_i, \omega_{y_i}), (p'_i, \omega'_{p_i}, y'_i, \omega'_{y_i})$ from the previous step, we first assert that $(p_i, \omega_{p_i}) = (p'_i, \omega'_{p_i})$. If this assertion fails, then we have an adversary $\mathcal{B}$ against DL-REL since $C_{p_i} = \mathbf{g}^{p_i} \cdot h^{\omega_{p_i}} = \mathbf{g}^{p'_i} \cdot h^{\omega'_{p_i}}$. Next, we can solve for $e_{i-1}, e_i, \omega_{e_{i-1}}, \omega_{e_i}$ from the linear equations
$$\begin{cases} y_i = w_{i,1} \cdot e_{i-1} + w_{i,2} \cdot e_i \\ y'_i = w'_{i,1} \cdot e_{i-1} + w'_{i,2} \cdot e_i \end{cases} \quad \text{and} \quad \begin{cases} \omega_{y_i} = w_{i,1} \cdot \omega_{e_{i-1}} + w_{i,2} \cdot \omega_{e_i} \\ \omega'_{y_i} = w'_{i,1} \cdot \omega_{e_{i-1}} + w'_{i,2} \cdot \omega_{e_i} \end{cases}.$$
Recall that we also have $\langle p_i, \mathbf{a}_i \rangle = y_i$ and $\langle p_i, \mathbf{a}'_i \rangle = y'_i$; taking the same linear combination used to solve the equations above would give us $p_i(0) + p_i(1) = e_{i-1}$ and $p_i(r_i) = e_i$. Thus, we have extracted valid polynomials for the information-theoretic sumcheck protocol.

□

Putting together the above special soundness results for the subprotocols, we obtain special soundness for Spartan-NIZK.

**Lemma 5.6.** Spartan-NIZK *satisfies* $\mathbf{n}$*-computational special soundness, where*
$$\mathbf{n} = (1, (1, 2_{\mathsf{li}}, 2)^{\mu}, 2, 2, 2, 1, (2, 2_{\mathsf{li}}, 2)^{\mu}, (\underbrace{4_{\pm}, \ldots, 4_{\pm}}_{\mu/2}, 2), 2).$$
*Concretely, there exists a PPT tree extractor $\mathcal{TE}_{\mathsf{Spartan\text{-}NIZK}}$ such that for every EPT adversary $\mathcal{A}$ against* SS *of* Spartan-NIZK*, there exists an EPT adversary $\mathcal{B}$ against* DL-REL*, as efficient as $\mathcal{A}$ and $\mathcal{TE}_{\mathsf{Spartan\text{-}NIZK}}$ combined, such that*
$$\mathbf{Adv}^{\mathsf{SS}}_{\mathsf{Spartan\text{-}NIZK},\mathbf{n}}(\mathcal{TE}_{\mathsf{Spartan\text{-}NIZK}}, \mathcal{A}) \leq \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, \sqrt{m}+2}(\mathcal{B}) + \frac{6\mu + 1}{|\mathbb{F}|}.$$

*Proof.* We describe the tree extractor $\mathcal{TE}_{\mathsf{Spartan\text{-}NIZK}}$. Given a $\mathbf{n}$-tree of accepting transcripts, it runs the following sub-extractors for the corresponding sub-trees:

1. Run $\mathcal{TE}_{\mathsf{SC}}$ for the first sumcheck subprotocol on each $(1, 2_{\mathsf{li}}, 2)^{\mu}$ sub-tree to extract polynomials $p_i(X)$ for $i \in [\mu]$ that satisfy the information-theoretic sumcheck protocol.

2. Run $\mathcal{TE}_{\mathsf{ProdPf}}, \mathcal{TE}_{\mathsf{OpenPf}}, \mathcal{TE}_{\mathsf{EqPf}}$ on each corresponding 2-subtree to extract claims $v_A, v_B, v_C$ such that $e_x = (v_A \cdot v_B - v_C) \cdot \widetilde{\mathrm{eq}}(r_x, \tau)$.
3. Run $\mathcal{TE}_{\mathsf{SC}}$ for the second sumcheck subprotocol on each $(1, 2_{\mathsf{li}}, 2)^\mu$ sub-tree to extract polynomials $p_i(X)$ for $i \in [\mu]$ that satisfy the information-theoretic sumcheck protocol.
4. Run $\mathcal{TE}_{\mathsf{PC}_{\mathsf{Multi}}}$ for the opening argument $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$ on the $(\underbrace{4_{\pm}, \ldots, 4_{\pm}}_{\mu/2}, 2)$ sub-tree, and on $2^{\mu/2} = \sqrt{m}$ different challenges $r_y$ provided by the $(2, 2_{\mathsf{li}}, 2)^\mu$ sub-tree, to extract a multilinear polynomial $\widetilde{w}(X)$ along with a correct evaluation $\widetilde{w}(r_y) = v_w$.
5. Run $\mathcal{TE}_{\mathsf{EqPf}}$ for the final equality proof to verify the equality $e_y = (r_A \cdot v_A + r_B \cdot v_B + r_C \cdot v_C) \cdot v_Z$.
6. Output the R1CS witness $w$.

Note that the $(2, 2_{\mathsf{li}}, 2)^\mu$ sub-tree in the second sumcheck subprotocol is necessary for extracting both from sumcheck, as well as from $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}$. We now consider the following hybrids. $\mathsf{Hyb}_0$ corresponds to the game $\mathsf{SS}$ for $\mathsf{Spartan\text{-}NIZK}$ with the tree extractor constructed above. $\mathsf{Hyb}_1$ is the same as $\mathsf{Hyb}_0$, but we additionally reject if the extracted R1CS witness is not satisfying. Conditioned on the event that none of the sub-extractor fails (and when that happens we get a $\mathsf{DL\text{-}REL}$ adversary $\mathcal{B}$), $\mathsf{Hyb}_1$ differs from $\mathsf{Hyb}_0$ exactly when the soundness of $\mathsf{Spartan\text{-}Core}$ is violated, which happens with probability at most $\frac{6\mu+1}{|\mathbb{F}|}$. □

Using Lemma 4.6 with Lemma 5.6, we conclude that $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ satisfies knowledge soundness. Note that the minimum partition size in the $\mathbf{n}$-tree of transcripts is $C = \frac{|\mathbb{F}|-1}{2}$.

**Theorem 5.7.** $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ *satisfies knowledge soundness. In particular, there exists an extractor* $\mathcal{E}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}$ *such that for every PPT prover* $\mathcal{P}^*$ *against* $\mathsf{KS}$ *of* $\mathsf{Spartan\text{-}NIZK}$ *making at most* $Q$ *random oracle queries, there exists an EPT adversary* $\mathcal{B}$ *against* $\mathsf{DL\text{-}REL}$ *such that*

$$\mathbf{Adv}^{\mathsf{KS}}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}(\mathcal{E}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}, \mathcal{P}^*) \leq \frac{Q(Q-1) + (Q+1)(13\mu+10) + 2(6\mu+1)}{|\mathbb{F}| - 1} + \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, \sqrt{m}+2}(\mathcal{B}).$$

*Here* $\mu = \log m$. *Both* $\mathcal{B}$ *and the extractor* $\mathcal{E}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}$ *runs in expected time that is at most* $O(Q \cdot m^6)$ *the running time of* $\mathcal{P}^*$.

Our next task is to exhibit a $k$-$\mathsf{ZK}$ simulator for $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$. The high-level idea is to let the simulator execute all subprotocols except the last with valid witnesses, then only invoke the simulator for the final $\mathsf{EqPf}$.

**Theorem 5.8.** $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ *satisfies* $(\mathsf{r}-1)$-$\mathsf{ZK}$, *where* $\mathsf{r} = 7\mu + 11$ *is the number of rounds of* $\mathsf{Spartan\text{-}NIZK}$.

*Proof.* See Figure 8 for a pseudocode description of our simulators. Using the sumcheck sub-simulator $\mathcal{S}_{\mathsf{SC}_{\mathsf{FS}}}$ in the top of the figure, we build the full simulator

$\mathcal{S}_{\mathsf{FS},\mathsf{r}-1}$ for $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$. From the construction of $\mathcal{S}_{\mathsf{FS},\mathsf{r}-1}$, it is clear that the proofs produced are accepting; this is because all the verifier's checks are done by checking the various proofs, which are either honestly generated, in which case validity follows from completeness, or by invoking the simulator, in which case validity follows from NIZK guarantee. Furthermore, $\mathcal{S}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}},\mathsf{r}-1}$ only makes a *single* RO reprogramming, which when the simulator $\mathcal{S}_{\mathsf{EqPf}_{\mathsf{FS}}}$ is invoked.

It remains to show that the output is indistinguishable from that of real transcripts. For the subprotocols, namely the $\Sigma$-protocols along with $\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}_{\mathsf{FS}}$, that we generate transcripts by generating honest proofs, we argue that they are indistinguishable. Firstly, the inputs to the arguments are the same (being perfectly blinded commitment). Secondly, the sub-protocols themselves are zero-knowledge, which implies witness indistinguishability. This further implies that the honestly generated proofs made by our simulator are identically distributed as proofs in real transcripts. In the last sub-protocol $\mathsf{EqPf}_{\mathsf{FS}}$ for which we use the simulator, we argue indistinguishability using the guarantee of the simulator $\mathcal{S}_{\mathsf{EqPf}_{\mathsf{FS}}}$. This concludes the proof of $k$-ZK. □

**Lemma 5.9.** $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ *satisfies perfect* $(\mathsf{r}-1)$-UR.

*Proof.* The last two rounds of $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ consists of an instance of the $\Sigma$-protocol $\mathsf{EqPf}_{\mathsf{FS}}$, which itself satisfies perfect 1-UR. In more detail, the last message in $\mathsf{EqPf}_{\mathsf{FS}}$ must be the unique scalar $z$ that satisfies $h^z = (C_1/C_2)^c \cdot \alpha$, where $C_1, C_2, \alpha$ are group elements determined by the previous messages. Hence $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ satisfies perfect $(\mathsf{r}-1)$-UR. □

Combining our results above, we obtain SIM-EXT for $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$.

**Theorem 5.10.** $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ *is simulation-extractable. Concretely, there exists a simulator-extractor* $\mathcal{E}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}$ *such that for every PPT adversary* $\mathcal{P}^*$ *against* SIM-EXT*, there exists an EPT adversary* $\mathcal{B}$ *against* DL-REL *such that*

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{SIM\text{-}EXT}}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}},\mathcal{R}_{\mathsf{R1CS}}}(\mathcal{S}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}},k}, \mathcal{E}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}, \mathcal{P}^*) \\ &\qquad \leq \frac{Q(Q-1)+(Q+1)(13\mu+10)+2(6\mu+1)+2}{|\mathbb{F}|-1} + \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G},\sqrt{m}+2}(\mathcal{B}).\end{aligned}$$

*Both* $\mathcal{B}$ *and* $\mathcal{E}_{\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}}$ *runs in expected time at most* $O(Q \cdot m^6)$ *that of* $\mathcal{P}^*$.

### 5.3 SIM-EXT of Spartan-SNARK

For Spartan-SNARK, the proof of SIM-EXT is similar to that of Spartan-NIZK. In particular, the proofs of $k$-ZK and $k$-UR carries over, and we only need to modify the proof of special soundness to accommodate for the more complex sparse multilinear polynomial commitment scheme. In what follows, we let $\mathsf{r}' = 7\mu + 11 + O(\mu)$ be the round complexity of Spartan-SNARK.

**Lemma 5.11.** $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$ *satisfies* $k$-ZK*, where* $k = \mathsf{r}' - 1$.

**Sub-simulator $\mathcal{S}_{\mathsf{SC}_{\mathsf{FS}}}$.**
**Input.** Public parameters $\mathsf{pp}$, a commitment $C_e$ to some value $e$ allegedly equal to $\sum_{x\in\{0,1\}^\mu} p(x)$, with $p$ a multivariate polynomial of individual degree at most $d$, and previous transcript $\mathsf{tr}$ (including $\mathsf{pp}$, R1CS input $(A, B, C, \mathsf{io})$, and all prover's messages so far).

Set $e_0 = e$. For $i = 1, \ldots, \mu$:

1. Sample $p_i(X) \xleftarrow{\$} \mathbb{F}^{\leq d}[X]$ randomly conditioned on $p_i(0) + p_i(1) = e_{i-1}$. Compute a commitment $C_{p_i} \leftarrow \mathsf{Commit}(\mathsf{pp}, p_i; \omega_{p_i})$ and append $C_{p_i}$ to $\mathsf{tr}$.
2. Obtain challenge $r_i \leftarrow \mathsf{H}(\mathsf{tr})$.
3. Let $e_i = p_i(r_i)$, compute a commitment $C_{e_i} \leftarrow \mathsf{Commit}(\mathsf{pp}, e_i; \omega_{e_i})$, and append $C_{e_i}$ to $\mathsf{tr}$.
4. Obtain challenges $w_{i,1}, w_{i,2} \leftarrow \mathsf{H}(\mathsf{tr})$.
5. Compute $\mathbf{a}, C_{y_i}, y_i, \omega_{y_i}$ as specified in the sumcheck subprotocol. Generate an honest proof $\pi_i \leftarrow \mathcal{P}_{\mathsf{DotProdPf}_{\mathsf{FS}}}(\mathsf{pp}, (C_{p_i}, C_{y_i}, \mathbf{a}_i), (p_i, \omega_{p_i}, y_i, \omega_{y_i}))$. Append $\pi_i$ to $\mathsf{tr}$.

After $\mu$ rounds, return $C_{e_\mu}$.

**Simulator $\mathcal{S}_{\mathsf{FS},\mathsf{r}-1}(\mathsf{pp}, (\mathbb{F}, A, B, C, \mathsf{io}))$:**

Initialize $\mathsf{tr} = (\mathsf{pp}, A, B, C, \mathsf{io})$.

1. Sample a random multilinear polynomial $\widetilde{w} \xleftarrow{\$} \mathbb{F}[X_1, \ldots, X_\mu]$. Compute $C_{\widetilde{w}} \leftarrow \mathsf{PC}_{\mathsf{Multi}}.\mathsf{Commit}(\mathsf{pp}, \widetilde{w}; \omega_{\widetilde{w}})$, and append $C_{\widetilde{w}}$ to $\mathsf{tr}$.
2. Obtain challenge $\tau \leftarrow \mathsf{H}(\mathsf{tr})$.
3. Run $\mathcal{S}_{\mathsf{SC}_{\mathsf{FS}}}$ on $(\mathsf{pp}, C_e)$ with current transcript $\mathsf{tr}$, and get output $C_{e_x} \leftarrow \mathsf{Commit}(\mathsf{pp}, e_x; \omega_{e_x})$ for some scalar $e_x \in \mathbb{F}$.
4. Sample $v_A, v_B, v_C \xleftarrow{\$} \mathbb{F}$ at random conditioned on $(v_A \cdot v_B - v_C) \cdot \widetilde{\mathrm{eq}}(r_x, \tau) = e_x$, and set $v_{AB} = v_A \cdot v_B$. Compute $C_{v_M} \leftarrow \mathsf{Commit}(\mathsf{pp}, v_M; \omega_M)$ for $M \in \{A, B, C\}$ along with $C_{v_{AB}} \leftarrow \mathsf{Commit}(\mathsf{pp}, v_{AB}; \omega_{AB})$, and append them to $\mathsf{tr}$.
5. Generate an honest proof

$$\pi_{\mathsf{ProdPf}} \leftarrow \mathcal{P}_{\mathsf{ProdPf}_{\mathsf{FS}}}(\mathsf{pp}, (C_{v_A}, C_{v_B}, C_{v_{AB}}), (v_A, v_B, \omega_{v_A}, \omega_{v_B}, \omega_{v_{AB}})),$$

and append it to $\mathsf{tr}$.
6. Generate an honest proof $\pi_{\mathsf{OpenPf}} \leftarrow \mathcal{P}_{\mathsf{OpenPf}_{\mathsf{FS}}}(\mathsf{pp}, (C_{v_C}), (v_C, \omega_{v_C}))$, and append it to $\mathsf{tr}$.
7. Generate an honest proof $\pi_{\mathsf{EqPf},1} \leftarrow \mathcal{P}_{\mathsf{EqPf}_{\mathsf{FS}}}(\mathsf{pp}, (C_{e_x}, C_{v'}), (\omega_{e_x} - \omega_{v'}))$, where $v' = (v_A \cdot v_B - v_C) \cdot \widetilde{\mathrm{eq}}(r_x, \tau)$ and $C_{v'} = (C_{v_{AB}}/C_{v_C})^{\widetilde{\mathrm{eq}}(r_x,\tau)}$; then append it to $\mathsf{tr}$.
8. Obtain challenges $r_A, r_B, r_C \leftarrow \mathsf{H}(\mathsf{tr})$.
9. Compute $C_T = r_A \cdot C_{v_A} + r_B \cdot C_{v_B} + r_C \cdot C_{v_C}$. Run $\mathcal{S}_{\mathsf{SC}_{\mathsf{FS}}}$ on $(\mathsf{pp}, C_T, \mathsf{tr})$, obtaining output $C_{e_y} \leftarrow \mathsf{Commit}(\mathsf{pp}, e_y; \omega_{e_y})$.
10. Generate opening proof $\pi_{\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}} \leftarrow \mathcal{P}_{\mathsf{PC}_{\mathsf{Multi}}.\mathsf{Open}_{\mathsf{FS}}}(\mathsf{pp}, (C_{\widetilde{w}}, r_y), (\widetilde{w}, \omega_{\widetilde{w}}))$; at the end, get $C_{v_w} = \mathsf{Commit}(\mathsf{pp}, v_w; \omega_{v_w})$, where $v_w \leftarrow \widetilde{w}(r_y[1\ldots])$, and append it to $\mathsf{tr}$.
11. Compute $v_Z = (1 - r_y[0]) \cdot v_w + r_y[0] \cdot \widetilde{(\mathsf{io}, 1)}(r_y[1\ldots])$ and $C_{v_Z} = C_{v_w}^{1-(r_y)_0} \cdot C_{v_{\mathsf{io}}}^{(r_y)_0}$.
12. Compute $v_1 = \widetilde{A}(r_x, r_y), v_2 = \widetilde{B}(r_x, r_y), v_3 = \widetilde{C}(r_x, r_y)$. Generate a *simulated* proof $\pi_{\mathsf{EqPf},2} \leftarrow \mathcal{S}_{\mathsf{EqPf}_{\mathsf{FS}}}$ for the equality $e_y = (r_A \cdot v_1 + r_B \cdot v_2 + r_C \cdot v_3) \cdot v_Z$. Append the proof to $\mathsf{tr}$.

Return $\mathsf{tr}$.

Fig. 8: Simulators for proof of $k$-ZK for Spartan.

*Proof.* We minimally modify the $k$-ZK simulator of $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$ to also output opening proofs $\mathsf{PC}_{\mathsf{SparseMulti}}.\mathsf{Open}_{\mathsf{FS}}$ for $\widetilde{M}(r_x, r_y)$ with $M \in \{A, B, C\}$. Since $A, B, C$ are part of the public input, the simulator has full access to the matrices, and hence can produce the proofs honestly. □

**Lemma 5.12.** $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$ *satisfies perfect* $k$-UR, *where* $k = \mathsf{r}' - 1$.

*Proof.* Since $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$ ends with the same invocation of the equality proof $\mathsf{EqPf}_{\mathsf{FS}}$, we obtain the same result as Lemma 5.9. □

The proof of knowledge soundness for $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$ is similar to that of $\mathsf{Spartan\text{-}NIZK}_{\mathsf{FS}}$, except we further need to extract the polynomials involved in $\mathsf{PC}_{\mathsf{SparseMulti}}.\mathsf{Open}$. We prove the lemma below in the full version.

**Lemma 5.13.** $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$ *satisfies knowledge soundness. Concretely, there exists an extractor* $\mathcal{E}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}$ *such that for every PPT prover* $\mathcal{P}^*$ *against* KS *of* Spartan-SNARK *making at most* $Q$ *random oracle queries, there exists an EPT adversary* $\mathcal{B}$ *against* DL-REL *such that*

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{KS}}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}(\mathcal{E}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}, \mathcal{P}^*)\\ &\leq \frac{Q(Q-1) + (Q+1)(25\mu + 9\nu + 16) + 6(m+n) + O(\mu+\nu)}{|\mathbb{F}| - 1} + \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, \sqrt{m+n}+2}(\mathcal{B}).\end{aligned}$$

*Here* $\mu = \log m, \nu = \log n$. *Both* $\mathcal{B}$ *and the extractor* $\mathcal{E}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}$ *runs in expected time that is at most* $O(Q \cdot m^{7.5} \cdot (m+n)^3)$ *the running time of* $\mathcal{P}^*$.

Combining the results above, we obtain SIM-EXT for $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$.

**Theorem 5.14.** $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$ *satisfies* SIM-EXT. *Concretely, there exists a simulator-extractor* $\mathcal{E}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}$ *such that for every PPT adversary* $\mathcal{P}^*$ *against* SIM-EXT *of* $\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}$, *there exists an EPT adversary* $\mathcal{B}$ *against* DL-REL *with*

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{SIM\text{-}EXT}}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}, \mathcal{R}_{\mathsf{R1CS}}}(\mathcal{S}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}, k}, \mathcal{E}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}, \mathcal{P}^*)\\ &\leq \frac{Q(Q-1) + (Q+1)(25\mu + 9\nu + 16) + 6(m+n) + O(\mu+\nu)}{|\mathbb{F}| - 1} + \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, \sqrt{m+n}+2}(\mathcal{B}).\end{aligned}$$

$\mathcal{B}$ *and* $\mathcal{E}_{\mathsf{Spartan\text{-}SNARK}_{\mathsf{FS}}}$ *run in expected time* $O(Q \cdot m^{7.5} \cdot (m+n)^3)$ *that of* $\mathcal{P}^*$.

## 6 Simulation Extractability of Bulletproofs

In this section, we show that the Bulletproofs protocols in [16] satisfy SIM-EXT, without relying on the AGM. The authors of [16] introduced two protocols, an *aggregate range proof* BP-ARP and an *arithmetic circuit satisfiability proof* BP-ACSPf[4], with both building on an *inner product argument* BP-IPA.

[4] To keep the naming consistent with [16], we refer to them as *proofs* even though they are actually *arguments*.

### 6.1 Aggregate Range Proof

We give a full description of the aggregate range proof BP-ARP in Figure 9. The value $m$ is the number of committed values $v_i$, and $n$ is the bit length of the upper bound (i.e., we prove $v_i \in [0, 2^n - 1]$ for all $i \in [m]$). Following the same approach as in Section 5 for Spartan, we need to establish three properties of $\mathsf{BP\text{-}ARP}_{\mathsf{FS}}$: (1) knowledge soundness, (2) the existence of a $k$-ZK simulator, and (3) $k$-UR for the same round $k$. We begin with the proof of knowledge soundness, which is essentially a restatement of the original result from [16].

**Lemma 6.1.** $\mathsf{BP\text{-}ARP}_{\mathsf{FS}}$ *satisfies* $(m \cdot n, m+2, 3, 2, \underbrace{4_{\pm}, \ldots, 4_{\pm}}_{\log(m \cdot n)})$*-computational special soundness, and hence knowledge soundness. Concretely, there exists an extractor* $\mathcal{E}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}}$ *such that for every PPT adversary* $\mathcal{P}^*$ *against* KS *making at most* $Q$ *random oracle queries, there exists an adversary* $\mathcal{A}$ *against* DL-REL *with*

$$\mathbf{Adv}^{\mathsf{KS}}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}}(\mathcal{E}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}}, \mathcal{P}^*) \leq \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, 2mn+3}(\mathcal{A}) + \frac{Q(Q-1) + 2(Q+1)\left(m(n+1) + 3\log(m \cdot n) + 3\right)}{|\mathbb{F}| - 1}.$$

*Both* $\mathcal{A}$ *and the extractor* $\mathcal{E}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}}$ *run in expected time that is at most* $O(Q \cdot m^4 \cdot n^3)$ *times the runtime of* $\mathcal{P}^*$*.*

*Proof.* The description of a tree extractor $\mathcal{TE}_{\mathsf{BP\text{-}ARP}}$, which either outputs a valid witness or a discrete log relation, can be found in [16]. This concludes the proof of computational special soundness. Combining Theorem 4.4 with Lemma 4.6, we conclude knowledge soundness for $\mathsf{BP\text{-}ARP}_{\mathsf{FS}}$. The expected runtime of the extractor $\mathcal{E}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}}$, as well as the adversary $\mathcal{A}$, is at most $O(Q \cdot (mn) \cdot (m+2) \cdot 6 \cdot (mn)^2) = O(Q \cdot m^4 \cdot n^3)$ times the runtime of $\mathcal{P}^*$, by Theorem 4.4. □

**Lemma 6.2.** $\mathsf{BP\text{-}ARP}_{\mathsf{FS}}$ *satisfies perfect* 2*-ZK.*

*Proof.* We present the 2-ZK simulator $\mathcal{S}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}, x}$ in Figure 10, and argue that its output is identically distributed to the output of an honest prover. All the challenges are chosen randomly as with real proofs. Next, in both real and simulated proofs, the proof elements $A, T_1, \beta_x, \mu$ and the underlying vectors $\mathbf{l}, \mathbf{r}$ are distributed uniformly among their respective domains. The proof elements $S, T_2$ are then uniquely determined from the previous ones from the verification equations that they must satisfy. Finally, both the scalar $\hat{t}$ and the inner product argument $\pi_{\mathsf{BP\text{-}IPA}}$ is generated deterministically from $\mathbf{l}, \mathbf{r}$; this implies identical distributions for those proof elements as well. □

Finally, we show the 2-UR property of BP-ARP. This result relies on the fact that $\mathsf{BP\text{-}IPA}_{\mathsf{FS}}$ has computationally unique proofs, shown in the full version.

**Lemma 6.3.** $\mathsf{BP\text{-}ARP}_{\mathsf{FS}}$ *satisfies* 2-UR*. In particular, for any adversary* $\mathcal{A}$ *against* 2-UR *of* $\mathsf{BP\text{-}ARP}_{\mathsf{FS}}$*, there exists an adversary* $\mathcal{B}$ *against* DL-REL *such that*

$$\mathbf{Adv}^{\text{2-UR}}_{\mathsf{BP\text{-}ARP}_{\mathsf{FS}}}(\mathcal{A}) \leq 2 \cdot \frac{Q(Q-1) + 6(Q+1)\log mn}{|\mathbb{F}| - 1} + 3 \cdot \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, 2mn+3}(\mathcal{B}) \ .$$

**Aggregate Range Proof Relation.**

$$\mathcal{R}_{\mathsf{BP\text{-}ARP}} = \left\{ \begin{array}{c} ((m, n, \mathbf{g}, \mathbf{h}, g, h, u), \mathbf{V}, (\mathbf{v}, \boldsymbol{\gamma})) : \\ V_j = g^{v_j} h^{\gamma_j} \ \wedge \ v_j \in [0, 2^n - 1] \ \forall j \in [1, m] \end{array} \right\}.$$

**Interaction Phase.** Denote $\mathbf{y}^{m \cdot n} = (1, y, \ldots, y^{m \cdot n - 1}) \in \mathbb{F}^{m \cdot n}$.

1. $\mathcal{P}$ samples $\alpha, \rho \xleftarrow{\$} \mathbb{F}$, $\mathbf{s}_L, \mathbf{s}_R \xleftarrow{\$} \mathbb{F}^{m \cdot n}$ and computes

$$\begin{aligned} &\mathbf{a}_L \in \{0,1\}^{m \cdot n} \text{ such that } \langle (\mathbf{a}_L)_{[(j-1)n, jn-1]}, \mathbf{2}^n \rangle = v_j \ \forall j \in [1, m], \\ &\mathbf{a}_R = \mathbf{a}_L - \mathbf{1}^{m \cdot n}, \\ &A = h^{\alpha} \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R}, \quad S = h^{\rho} \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}. \end{aligned}$$

$\mathcal{P}$ sends $A, S$ to $\mathcal{V}$.

2. $\mathcal{V}$ sends challenges $y, z \xleftarrow{\$} \mathbb{F}^*$.

3. $\mathcal{P}$ samples $\beta_1, \beta_2 \xleftarrow{\$} \mathbb{F}$ and computes

$$\begin{aligned} &\ell(X) = (\mathbf{a}_L - z \cdot \mathbf{1}^{m \cdot n}) + \mathbf{s}_L \cdot X, \\ &r(X) = \mathbf{y}^{m \cdot n} \circ (\mathbf{a}_R + z \cdot \mathbf{1}^{m \cdot n} + \mathbf{s}_R \cdot X) + \sum_{j=1}^{m} z^{j+1} \cdot \left( \mathbf{0}^{(j-1)n} \| \mathbf{2}^n \| \mathbf{0}^{(m-j)n} \right), \\ &t(X) = \langle \ell(X), r(X) \rangle = t_0 + t_1 \cdot X + t_2 \cdot X^2, \quad T_1 = g^{t_1} h^{\beta_1}, \quad T_2 = g^{t_2} h^{\beta_2}. \end{aligned}$$

$\mathcal{P}$ sends $T_1, T_2$ to $\mathcal{V}$.

4. $\mathcal{V}$ sends challenge $x \xleftarrow{\$} \mathbb{F}^*$.

5. $\mathcal{P}$ computes

$$\begin{aligned} &\mathbf{l} = \ell(x), \quad \mathbf{r} = r(x), \quad \hat{t} = \langle \mathbf{l}, \mathbf{r} \rangle, \quad \mu = \alpha + \rho \cdot x, \\ &\beta_x = \beta_2 \cdot x^2 + \beta_1 \cdot x + \sum_{j=1}^{m} z^{j+1} \cdot \gamma_j. \end{aligned}$$

$\mathcal{P}$ sends $\hat{t}, \beta_x, \mu$ to $\mathcal{V}$.

6. $\mathcal{V}$ sends challenge $w \xleftarrow{\$} \mathbb{F}^*$.

7. $\mathcal{P}, \mathcal{V}$ both compute

$$\begin{aligned} &\mathbf{h}' = \mathbf{h}^{\mathbf{y}^{-m \cdot n}}, \quad u' = u^w, \\ &P' = h^{-\mu} \cdot A \cdot S^x \cdot \mathbf{g}^{-z \cdot \mathbf{1}^{m \cdot n}} \cdot (\mathbf{h}')^{z \cdot \mathbf{y}^{m \cdot n}} \cdot \prod_{j=1}^{m} (\mathbf{h}')^{z^{j+1} \cdot \mathbf{2}^n}_{[(j-1)n, jn-1]} \cdot (u')^{\hat{t}}. \end{aligned}$$

8. $\mathcal{P}, \mathcal{V}$ engage in $\mathsf{BP\text{-}IPA}$ for the triple $((m \cdot n, \mathbf{g}, \mathbf{h}', u'), P', (\mathbf{l}, \mathbf{r}))$.

**Verification.**

1. $\mathcal{V}$ rejects if $\mathsf{BP\text{-}IPA}$ fails.
2. $\mathcal{V}$ computes $R = \mathbf{V}^{z^2 \cdot \mathbf{z}^m} \cdot g^{(z - z^2) \cdot \langle \mathbf{1}^{m \cdot n}, \mathbf{y}^{m \cdot n} \rangle - \sum_{j=1}^{m} z^{j+2} \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle} \cdot T_1^x \cdot T_2^{x^2}$.
3. $\mathcal{V}$ checks whether $g^{\hat{t}} h^{\beta_x} \stackrel{?}{=} R$.

Fig. 9: Bulletproofs' Aggregate Range Proof $\mathsf{BP\text{-}ARP}$

**Simulator** $\mathcal{S}_{\mathsf{BP\text{-}ARP_{FS}},x}(\mathsf{pp} = (m, n, \mathbf{g}, \mathbf{h}, g, h, u), \mathbf{V})$**:**

1. Initialize $\mathsf{tr} = (\mathsf{pp}, \mathbf{V})$. Sample $\alpha, \rho \xleftarrow{\$} \mathbb{F}$, $\mathbf{a}_L, \mathbf{a}_R, \mathbf{s}_L, \mathbf{s}_R \xleftarrow{\$} \mathbb{F}^{m \cdot n}$ and compute $A = h^{\alpha} \mathbf{g}^{\mathbf{a}_L} \mathbf{h}^{\mathbf{a}_R}$, $S = h^{\rho} \mathbf{g}^{\mathbf{s}_L} \mathbf{h}^{\mathbf{s}_R}$. Append $A, S$ to $\mathsf{tr}$.
2. Obtain challenges $y, z \leftarrow \mathsf{H}(\mathsf{tr})$.
3. Sample $x \xleftarrow{\$} \mathbb{F}^*$ and compute $\mu = \alpha + \rho \cdot x$, $\mathbf{l} = (\mathbf{a}_L - z \cdot \mathbf{1}^{m \cdot n}) + \mathbf{s}_L \cdot x$, $\mathbf{r} = \mathbf{y}^{m \cdot n} \circ (\mathbf{a}_R + z \cdot \mathbf{1}^{m \cdot n} + \mathbf{s}_R \cdot x) + \sum_{j=1}^{m} z^{j+1} \cdot \left(\mathbf{0}^{(j-1)n} \| \mathbf{2}^n \| \mathbf{0}^{(m-j)n}\right)$, and $\hat{t} = \langle \mathbf{l}, \mathbf{r} \rangle$.
4. Sample $\beta_x \xleftarrow{\$} \mathbb{F}$, $T_1 \xleftarrow{\$} \mathbb{G}$, and compute $T_2 = \left(g^{\hat{t} - \delta(y,z)} \cdot h^{\beta_x} \cdot \mathbf{V}^{-z^2 \cdot \mathbf{z}^m} \cdot T_1^{-x}\right)^{x^{-2}}$, where $\delta(y, z) = (z - z^2) \cdot \langle \mathbf{1}^{m \cdot n}, \mathbf{y}^{m \cdot n} \rangle - \sum_{j=1}^{m} z^{j+2} \cdot \langle \mathbf{1}^n, \mathbf{2}^n \rangle$. Append $T_1, T_2$ to $\mathsf{tr}$.
5. Reprogram $\mathsf{H}(\mathsf{tr}) := x$, then append $\hat{t}, \beta_x, \mu$ to $\mathsf{tr}$.
6. Obtain challenge $w \leftarrow \mathsf{H}(\mathsf{tr})$.
7. Compute $\mathbf{h}' = \mathbf{h}^{\mathbf{y}^{-m \cdot n}}$, $u' = u^w$, and
$$P' = h^{-\mu} \cdot A \cdot S^x \cdot \mathbf{g}^{-z \cdot \mathbf{1}^{m \cdot n}} \cdot (\mathbf{h}')^{z \cdot \mathbf{y}^{m \cdot n}} \cdot \prod_{j=1}^{m} (\mathbf{h}')^{z^{j+1} \cdot \mathbf{2}^n}_{[(j-1)n, jn-1]} \cdot (u')^{\hat{t}}.$$
8. Generate an honest proof $\pi_{\mathsf{BP\text{-}IPA_{FS}}} \leftarrow \mathcal{P}_{\mathsf{BP\text{-}IPA_{FS}}}((m \cdot n, \mathbf{g}, \mathbf{h}', u'), P', (\mathbf{l}, \mathbf{r}))$.
9. Output $\pi_{\mathsf{BP\text{-}ARP_{FS}}} = (A, S, T_1, T_2, \hat{t}, \beta_x, \mu, \pi_{\mathsf{BP\text{-}IPA_{FS}}})$.

Fig. 10: $\mathsf{BP\text{-}ARP_{FS}}$ $k$-$\mathsf{ZK}$ simulator

*$\mathcal{B}$ runs in expected time at most $O(Q \cdot m^2 \cdot n^2)$ that of $\mathcal{A}$'s runtime.*

*Proof.* We proceed through a sequence of hybrids. The high-level idea is to analyze different cases for where the two proofs $\pi, \pi'$ first differ after the $x$ challenge, and reduce each case to breaking $\mathsf{DL\text{-}REL}$ or the unique proof property of $\mathsf{BP\text{-}IPA}$ (which in turn reduces to breaking $\mathsf{DL\text{-}REL}$).

- $\mathsf{Hyb}_0$ is the game $\mathsf{2\text{-}UR}^{\mathcal{A}}_{\mathsf{BP\text{-}ARP_{FS}}}$. Recall that in this game, an adversary $\mathcal{A}$ outputs an input $\mathbf{V}$, a challenge $x \in \mathbb{F}^*$, and two proofs $\pi, \pi'$ that agrees up to the $x$ challenge, i.e. we have $\pi = (A, S, T_1, T_2, \hat{t}, \beta_x, \mu, \pi_{\mathsf{BP\text{-}IPA_{FS}}})$ and $\pi' = (A, S, T_1, T_2, \hat{t}', \beta'_x, \mu', \pi'_{\mathsf{BP\text{-}IPA_{FS}}})$. $\mathcal{A}$ wins if $\pi \neq \pi'$ and both proofs are accepting with respect to the $x$ challenge that it chose.
- $\mathsf{Hyb}_1$ is the same as $\mathsf{Hyb}_0$, except that we also run $\mathcal{E}_{\mathsf{BP\text{-}IPA_{FS}}}$ on the proofs $\pi_{\mathsf{BP\text{-}IPA_{FS}}}, \pi'_{\mathsf{BP\text{-}IPA_{FS}}}$ to extract witnesses $(\mathbf{l}, \mathbf{r})$ and $(\mathbf{l}', \mathbf{r}')$. $\mathsf{Hyb}_1$ returns 0 if the extractor aborts on either proofs, or $\hat{t} \neq \langle \mathbf{l}, \mathbf{r} \rangle$ or $\hat{t}' \neq \langle \mathbf{l}', \mathbf{r}' \rangle$.

We can see that $\mathsf{Hyb}_1$ is identical to $\mathsf{Hyb}_0$, except when the extractor $\mathcal{E}_{\mathsf{BP\text{-}IPA_{FS}}}$ fails in extracting from either proofs $\pi_{\mathsf{BP\text{-}IPA_{FS}}}, \pi'_{\mathsf{BP\text{-}IPA_{FS}}}$. The probability that this happens is precisely bounded by twice the $\mathsf{KS}$ advantage of $\mathsf{BP\text{-}IPA_{FS}}$. Concretely, invoking the extractor for $\mathsf{BP\text{-}IPA_{FS}}$, there exists an adversary $\mathcal{B}$ against $\mathsf{DL\text{-}REL}$, running in expected time at most $O(Q \cdot m^2 \cdot n^2)$ that of $\mathcal{A}$'s runtime, such that

$$|\Pr[\mathsf{Hyb}_0] - \Pr[\mathsf{Hyb}_1]| \leq 2\frac{Q(Q-1) + 6(Q+1)\log(m \cdot n)}{|\mathbb{F}| - 1} + 2\mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, 2mn+3}(\mathcal{B}) \ .$$

It remains to show that if $\mathsf{Hyb}_1$ returns 1, then there exists an adversary $\mathcal{B}'$ that returns a non-trivial discrete log relation. Adversary $\mathcal{B}'$ is as follows:

- **If $\hat{t} \neq \hat{t}'$ or $\beta_x \neq \beta'_x$:** since both proofs are accepting and are the same up to the $x$ challenge, we have
$$g^{\hat{t}} \cdot h^{\beta_x} = V^{z^2} \cdot g^{\delta(y,z)} \cdot T_1^x \cdot T_2^{x^2} = g^{\hat{t}'} \cdot h^{\beta'_x}.$$

- **If $(\hat{t}, \beta_x) = (\hat{t}', \beta'_x)$ but $\mu \neq \mu'$:** since both proofs $\pi_{\mathsf{BP\text{-}IPA_{FS}}}, \pi'_{\mathsf{BP\text{-}IPA_{FS}}}$ are accepting, we have

$$\begin{aligned}\mathbf{g}^{\mathbf{l}} \cdot \mathbf{h}^{(\mathbf{y}^{-m\cdot n} \circ \mathbf{r})} \cdot h^{\mu} &= A \cdot S^{x} \cdot \mathbf{g}^{-z\cdot \mathbf{1}^{m\cdot n}} \cdot (\mathbf{h}')^{z\cdot \mathbf{y}^{m\cdot n}} \cdot \prod_{j=1}^{m} (\mathbf{h}')^{z^{j+1}\cdot \mathbf{2}^{n}}_{[(j-1)n, jn-1]} \cdot u^{w\cdot \hat{t}} \\ &= \mathbf{g}^{\mathbf{l}'} \cdot \mathbf{h}^{(\mathbf{y}^{-m\cdot n} \circ \mathbf{r}')} \cdot h^{\mu'}.\end{aligned}$$

- **If $(\hat{t}, \beta_x, \mu) = (\hat{t}', \beta'_x, \mu')$ but $\pi_{\mathsf{BP\text{-}IPA_{FS}}} \neq \pi'_{\mathsf{BP\text{-}IPA_{FS}}}$:** here, we know that both $\mathsf{BP\text{-}IPA_{FS}}$ proofs are for the same statement $P'$, with extracted witnesses $(\mathbf{l}, \mathbf{r})$, $(\mathbf{l}', \mathbf{r}')$. From the proof that $\mathsf{BP\text{-}IPA_{FS}}$ is computationally unique, the two witnesses must be different, which gives a discrete log relation.

Note that the first two cases above give discrete log relations, and if $\mathsf{Hyb}_1$ returns 1, then $\pi \neq \pi'$, hence at least one of the above cases happens. Putting everything together and unifying $\mathcal{B}, \mathcal{B}'$ we get the desired bound. □

We finally obtain SIM-EXT from the previous results and Theorem 3.4.

**Theorem 6.4.** $\mathsf{BP\text{-}ARP_{FS}}$ *satisfies* SIM-EXT. *In particular, there exists a simulator-extractor* $\mathcal{E}_{\mathsf{BP\text{-}ARP_{FS}}}$ *such that for any adversary* $\mathcal{P}^*$ *against* SIM-EXT *of* $\mathsf{BP\text{-}ARP_{FS}}$, *there exists an adversary* $\mathcal{B}$ *against* DL-REL *such that*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{SIM\text{-}EXT}}_{\mathsf{BP\text{-}ARP_{FS}}}(\mathcal{E}_{\mathsf{BP\text{-}ARP_{FS}}}, \mathcal{P}^*) \leq\ & 4 \cdot \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, 2mn+3}(\mathcal{B}) \\ & + \frac{3Q(Q-1) + 2(Q+1)\left(m(n+1) + 6\log(mn) + 3\right) + 2}{|\mathbb{F}| - 1}.\end{aligned}$$

*$\mathcal{B}$ runs in expected time at most $O(Q \cdot m^4 \cdot n^3)$ the runtime of $\mathcal{P}^*$.*

### 6.2 Arithmetic Circuit Satisfiability Proof

We will describe BP-ACSPf and prove the following theorem in the full version.

**Theorem 6.5.** $\mathsf{BP\text{-}ACSPf_{FS}}$ *satisfies* SIM-EXT. *Concretely, there exists a simulator-extractor* $\mathcal{E}_{\mathsf{BP\text{-}ACSPf_{FS}}}$ *such that for any adversary* $\mathcal{P}^*$ *against* SIM-EXT *of* $\mathsf{BP\text{-}ACSPf_{FS}}$, *there exists an adversary* $\mathcal{B}$ *against* DL-REL *such that*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{SIM\text{-}EXT}}_{\mathsf{BP\text{-}ACSPf_{FS}}}(\mathcal{E}_{\mathsf{BP\text{-}ACSPf_{FS}}}, \mathcal{P}^*) \leq & 4 \cdot \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G}, 2n+1}(\mathcal{B}) \\ & + \frac{3Q(Q-1) + 2(Q+1)(n+q+9\log n+6) + 2}{|\mathbb{F}| - 1}.\end{aligned}$$

*Here $n$ is the number of multiplication gates, and $q$ is the number of committed inputs. $\mathcal{B}$ runs in expected time at most $O(Q \cdot q \cdot n^3)$ the runtime of $\mathcal{P}^*$.*

## 7 Quantitative discussion of our SIM-EXT bounds

In this section, we briefly show how to interpret the tightness of our SIM-EXT bounds for Bulletproofs and Spartan, and compare them with the previous analyses of [30,33] using AGM. We leave a detailed discussion to the full version.

By Theorem 3.4, we see that the SIM-EXT advantage is tightly related to the KS advantage. Thus, we will compare the latter. For BP-ARP with $m = 1$ (range proof of a single value), we compare our KS bound with the AGM-based bound of [33] in Figure 11. Our approach loses tightness due to two factors: first, we lose a factor of $Q$ due to rewinding (shown to be somewhat inherent for the similar case of Schnorr signatures [26]), and second, our DL-REL adversary is *expected time*, which leads to another "square-root" loss in security [42] (namely $\mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G},2n+3}(\mathcal{A}) \leq \sqrt{t(\mathcal{A})^2/|\mathbb{F}|}$ for generic attacks). Our concrete KS advantages for Spartan is even lower, due to the bigger tree sizes of Spartan. We leave achieving tighter rewinding-based bounds to future work.

| | **Lemma 6.1** | [33, **Theorem 4**] |
|---|---|---|
| *Asymptotic* | $O\left(\frac{Q^2+Qn}{\lvert\mathbb{F}\rvert}\right) + \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G},2n+3}(\mathcal{A})$ where $\mathbb{E}[t(\mathcal{A})] = O(Q \cdot n^3 \cdot t(\mathcal{P}^*))$ | $O\left(\frac{Qn}{\lvert\mathbb{F}\rvert}\right) + \mathbf{Adv}^{\mathsf{DL\text{-}REL}}_{\mathbb{G},2n+3}(\mathcal{A}')$ where $t(\mathcal{A}') = O(Q \cdot n)$ |
| *Concrete* | $\approx 22$ bits of security | $\approx 164$ bits of security |

Fig. 11: Comparison of KS advantages, obtained by rewinding (ours) versus AGM [33], for Bulletproofs' single range proof, e.g. BP-ARP with $m = 1$. Here $t(\cdot)$ denotes the running time. For concrete advantage, we take $|\mathbb{F}| \approx 2^{256}$, $n = 64$, $t(\mathcal{P}^*) = 2^{48}$, $Q = 2^{40}$.

## References


1. Aleo. https://www.aleo.org/ (2022)
2. Attema, T., Fehr, S., Klooß, M.: Fiat-shamir transformation of multi-round interactive proofs. Cryptology ePrint Archive, Report 2021/1377 (2021), https://eprint.iacr.org/2021/1377
3. Baghery, K., Kohlweiss, M., Siim, J., Volkhov, M.: Another look at extraction and randomization of groth's zk-SNARK. Cryptology ePrint Archive, Report 2020/811 (2020), https://eprint.iacr.org/2020/811
4. Baghery, K., Pindado, Z., Ràfols, C.: Simulation extractable versions of groth's zk-SNARK revisited. In: Krenn, S., Shulman, H., Vaudenay, S. (eds.) CANS 20. LNCS, vol. 12579, pp. 453–461. Springer, Heidelberg (Dec 2020)
5. Baghery, K., Sedaghat, M.: Tiramisu: Black-box simulation extractable NIZKs in the updatable CRS model. Cryptology ePrint Archive, Report 2020/474 (2020), https://eprint.iacr.org/2020/474
6. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (May / Jun 2006)
7. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046 (2018), https://eprint.iacr.org/2018/046
8. Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from bitcoin. In: 2014 IEEE Symposium on Security and Privacy. pp. 459–474. IEEE Computer Society Press (May 2014)
9. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for R1CS. In: Ishai, Y., Rijmen, V. (eds.)

EUROCRYPT 2019, Part I. LNCS, vol. 11476, pp. 103–128. Springer, Heidelberg (May 2019)
10. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 31–60. Springer, Heidelberg (Oct / Nov 2016)
11. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 626–643. Springer, Heidelberg (Dec 2012)
12. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for SNARKs and proof-carrying data. Cryptology ePrint Archive, Report 2012/095 (2012), https://eprint.iacr.org/2012/095
13. Boneh, D., Drake, J., Fisch, B., Gabizon, A.: Halo infinite: Proof-carrying data from additive polynomial commitments. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part I. LNCS, vol. 12825, pp. 649–680. Springer, Heidelberg, Virtual Event (Aug 2021)
14. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 327–357. Springer, Heidelberg (May 2016)
15. Bowe, S., Grigg, J., Hopwood, D.: Halo: Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Report 2019/1021 (2019), https://eprint.iacr.org/2019/1021
16. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy. pp. 315–334. IEEE Computer Society Press (May 2018)
17. Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK compilers. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 677–706. Springer, Heidelberg (May 2020)
18. Canetti, R., Sarkar, P., Wang, X.: Triply adaptive UC NIZK. Cryptology ePrint Archive, Report 2020/1212 (2020), https://eprint.iacr.org/2020/1212
19. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 78–96. Springer, Heidelberg (Aug 2006)
20. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.P.: Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 738–768. Springer, Heidelberg (May 2020)
21. Coda. https://codaprotocol.com/ (2022)
22. Decker, C., Wattenhofer, R.: Bitcoin transaction malleability and MtGox. In: Kutylowski, M., Vaidya, J. (eds.) ESORICS 2014, Part II. LNCS, vol. 8713, pp. 313–326. Springer, Heidelberg (Sep 2014)
23. Don, J., Fehr, S., Majenz, C.: The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 602–631. Springer, Heidelberg (Aug 2020)
24. Faust, S., Kohlweiss, M., Marson, G.A., Venturi, D.: On the non-malleability of the Fiat-Shamir transform. In: Galbraith, S.D., Nandi, M. (eds.) INDOCRYPT 2012. LNCS, vol. 7668, pp. 60–79. Springer, Heidelberg (Dec 2012)
25. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987)

26. Fleischhacker, N., Jager, T., Schröder, D.: On tight security proofs for Schnorr signatures. Journal of Cryptology 32(2), 566–599 (Apr 2019)
27. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 33–62. Springer, Heidelberg (Aug 2018)
28. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953 (2019), https://eprint.iacr.org/2019/953
29. Ganesh, C., Khoshakhlagh, H., Kohlweiss, M., Nitulescu, A., Zając, M.: What makes fiat–shamir zksnarks (updatable srs) simulation extractable? In: SCN (2022)
30. Ganesh, C., Orlandi, C., Pancholi, M., Takahashi, A., Tschudi, D.: Fiat-shamir bulletproofs are non-malleable (in the algebraic group model). In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 397–426. Springer, Heidelberg (May / Jun 2022)
31. Ganesh, C., Orlandi, C., Pancholi, M., Takahashi, A., Tschudi, D.: Fiat-shamir bulletproofs are non-malleable (in the random oracle model). Cryptology ePrint Archive (2023)
32. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer, Heidelberg (May 2013)
33. Ghoshal, A., Tessaro, S.: Tight state-restoration soundness in the algebraic group model. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part III. LNCS, vol. 12827, pp. 64–93. Springer, Heidelberg, Virtual Event (Aug 2021)
34. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: 17th ACM STOC. pp. 291–304. ACM Press (May 1985)
35. Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: Linear-time and post-quantum SNARKs for R1CS. Cryptology ePrint Archive, Report 2021/1043 (2021), https://eprint.iacr.org/2021/1043
36. Grin: a minimal implementation of mimblewimble. https://github.com/mimblewimble/grin (2022)
37. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (Dec 2006)
38. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 305–326. Springer, Heidelberg (May 2016)
39. Groth, J., Maller, M.: Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 581–612. Springer, Heidelberg (Aug 2017)
40. Grubbs, P., Arun, A., Zhang, Y., Bonneau, J., Walfish, M.: Zero-knowledge middleboxes. In: Butler, K.R.B., Thomas, K. (eds.) USENIX Security 2022. pp. 4255–4272. USENIX Association (Aug 2022)
41. Haines, T., Lewis, S.J., Pereira, O., Teague, V.: How not to prove your election outcome. In: 2020 IEEE Symposium on Security and Privacy. pp. 644–660. IEEE Computer Society Press (May 2020)
42. Jaeger, J., Tessaro, S.: Expected-time cryptography: Generic techniques and applications to concrete soundness. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 414–443. Springer, Heidelberg (Nov 2020)

43. Jain, A., Pandey, O.: Non-malleable zero knowledge: Black-box constructions and definitional relationships. In: Abdalla, M., Prisco, R.D. (eds.) SCN 14. LNCS, vol. 8642, pp. 435–454. Springer, Heidelberg (Sep 2014)
44. Kothapalli, A., Setty, S., Tzialla, I.: Nova: Recursive zero-knowledge arguments from folding schemes. Cryptology ePrint Archive, Report 2021/370 (2021), https://eprint.iacr.org/2021/370
45. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 2111–2128. ACM Press (Nov 2019)
46. Bulletproofs+ in monero. https://www.getmonero.org/2020/12/24/Bulletproofs+-in-Monero.html (2020)
47. Mt.Gox press release. https://web.archive.org/web/20140214041924/https://www.mtgox.com/press_release_20140210.html (2014)
48. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: Nearly practical verifiable computation. In: 2013 IEEE Symposium on Security and Privacy. pp. 238–252. IEEE Computer Society Press (May 2013)
49. Pass, R.: On deniability in the common reference string and random oracle model. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 316–337. Springer, Heidelberg (Aug 2003)
50. Pass, R., Rosen, A.: New and improved constructions of non-malleable cryptographic protocols. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 533–542. ACM Press (May 2005)
51. Polygon. https://polygon.technology/ (2022)
52. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: 40th FOCS. pp. 543–553. IEEE Computer Society Press (Oct 1999)
53. Scroll. https://scroll.io/ (2022)
54. Setty, S.: Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 704–737. Springer, Heidelberg (Aug 2020)
55. Setty, S., Lee, J.: Quarks: Quadruple-efficient transparent zkSNARKs. Cryptology ePrint Archive, Report 2020/1275 (2020), https://eprint.iacr.org/2020/1275
56. Starkware. https://starkware.co/ (2022)
57. Wahby, R.S., Tzialla, I., shelat, a., Thaler, J., Walfish, M.: Doubly-efficient zkSNARKs without trusted setup. In: 2018 IEEE Symposium on Security and Privacy. pp. 926–943. IEEE Computer Society Press (May 2018)
58. Wee, H.: Zero knowledge in the random oracle model, revisited. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 417–434. Springer, Heidelberg (Dec 2009)
59. Wikström, D.: Special soundness in the random oracle model. Cryptology ePrint Archive, Report 2021/1265 (2021), https://eprint.iacr.org/2021/1265
60. Xie, T., Zhang, Y., Song, D.: Orion: Zero knowledge proof with linear prover time. Cryptology ePrint Archive, Report 2022/1010 (2022), https://eprint.iacr.org/2022/1010
61. Zhang, F., Maram, D., Malvai, H., Goldfeder, S., Juels, A.: DECO: Liberating web data using decentralized oracles for TLS. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 1919–1938. ACM Press (Nov 2020)
62. ZKProofs Standards. https://zkproof.org/ (2022)