Critical-Round Special Soundness for Multi-Round Proofs

Masayuki Abe[1], David Balbás[2], Dung Bui[3], Miyako Ohkubo[4],
Zehua Shang[5], Akira Takahashi[6], and Mehdi Tibouchi[1]

[1] NTT Social Informatics Laboratories, Japan
abe.masayuki@iecl.ntt.co.jp, mehdi.tibouchi@ntt.com
[2] ETH Zurich, Switzerland dbalbasg@gmail.com
[3] Northwestern University, USA dung.bui@northwestern.edu
[4] NICT, Japan m.ohkubo@nict.go.jp
[5] Kyoto University & Yokohama National University, Japan zehuashang44@gmail.com
[6] J.P. Morgan, USA
takahashi.akira.58s@gmail.com

Abstract. Multi-round public-coin proof systems have progressed rapidly, yet their simulators and extractors are rarely used as building blocks in other protocols, unlike those of 3-move Σ-protocols. Prior work [Abe et al., Eurocrypt '26] addressed the simulator side with critical-round zero-knowledge.

In this paper, we address the extractor side and introduce *critical-round special soundness* (CRSS): a witness is extractable from a small *critical subtree* of transcripts that branches only from one round, while the full tree of transcripts serves only as a fall-back that outputs a solution to a hardness assumption. We first establish an extract-or-extend theorem that bounds the failure probability of the critical extractor even when the critical-round challenges are chosen by the adversary, and then show that KKW-style MPC-in-the-head proofs, KZG-compiled polynomial IOPs including Plonk, and Protogalaxy satisfy CRSS. We also give a corresponding tree-extraction result for the sumcheck sub-protocol of Spartan.

We present three applications: a hash-based witness sharing scheme that verifiably secret-shares an NP witness without interaction among recipients, the first offline trapdoor-extractable trapdoor commitment from a multi-round proof, where a trapdoor is extracted immediately upon a double opening, and a parameter improvement for the multi-round Fischlin transform [Rotem-Tessaro, Crypto '25] that reduces the prover's complexity in practice.

Keywords: Multi-Round Proofs, Critical Round, Special Soundness, Witness Sharing, Trapdoor Commitment, Fischlin Transform

Table of Contents

Critical-Round Special Soundness for Multi-Round Proofs . . . 1
Masayuki Abe, David Balbás, Dung Bui, Miyako Ohkubo, Zehua Shang, Akira Takahashi, and Mehdi Tibouchi
1 Introduction . . . 3
1.1 Our Contributions . . . 3
1.2 Technical Overview . . . 5
1.3 Related Work . . . 9
2 Preliminaries . . . 10
2.1 Public-Coin Proof System . . . 10
2.2 Fiat-Shamir Transformation with Open Round . . . 11
2.3 Predicate Special Soundness . . . 12
3 Critical-Round Special Soundness . . . 14
3.1 Definition of Critical-Round Special Soundness . . . 14
3.2 Extract-or-Extend Theorem for CRSS Proof Systems . . . 14
3.3 Instantiations of Critical-Round Proof Systems . . . 17
4 Witness Sharing . . . 19
4.1 Definitions . . . 19
4.2 Generic Construction from CRSS Proof System . . . 20
4.3 Construction from MPCitH . . . 22
4.4 Comparison with Alternative Approaches . . . 26
5 Offline Trapdoor-Extractable Trapdoor Commitments . . . 27
5.1 Definitions . . . 27
5.2 Construction from Multi-round ZK proof systems . . . 29
6 A CRSS Variant of the Multi-Round Fischlin Transform . . . 31
7 Conclusion . . . 36
A Omitted Preliminaries . . . 41
A.1 Zero-Knowledge Notions . . . 41
B Critical Round in Other Proof Systems . . . 41
B.1 Cut-and-choose KKW . . . 41
B.2 Σ-Protocols with Helper . . . 45
B.3 KZG-Compiled Polynomial IOP . . . 47
B.4 Sumcheck Sub-protocol of Spartan . . . 50
B.5 Protogalaxy . . . 52
C Katz-Kolesnikov-Wang Framework for MPC-in-the-Head with Preprocessing . . . 56
C.1 Bare-bones KKW . . . 57
C.2 Cut-and-Choose KKW . . . 57

1 Introduction

Multi-round public-coin proof systems are a prominent and increasingly popular tool in modern cryptography. In such protocols, a prover aims to convince a verifier of the validity of a statement x by interacting over multiple rounds. The prover first sends an initial message a, after which the verifier samples challenges $c_1, \ldots, c_\mu$ independently in successive rounds, and the prover responds with messages $z_1, \ldots, z_\mu$. The resulting transcript has the form $(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.

In the realm of multi-round interactive proofs, the gold standard for soundness is captured by the notion of *special soundness*. This framework ensures that, given a sufficient number of accepting transcripts with distinct challenges across rounds, one can efficiently extract a witness for the underlying statement. Specifically, the $(k_1, \ldots, k_\mu)$-*special soundness* framework generalizes this property to multi-round protocols, guaranteeing witness extraction from k_i accepting responses to distinct challenges at each round i, structured as an *accepting tree of transcripts* [ACK21, AFR23]. For arguments based on computational assumptions, extraction is defined with respect to an *extended transcript tree*: given such a tree, an extractor either outputs a witness or breaks an underlying hardness assumption [AFR23]. Naturally, $(k_1, \ldots, k_\mu)$-special soundness generalizes the special soundness of Σ-protocols [Cra96], and provides tight soundness analyses for general multi-round protocols, including non-interactive variants of these protocols when compiled via the Fiat-Shamir transform [FS87]. Zero-knowledge admits an analogous extension which also generalizes the usual notion for Σ-protocols. In multi-round honest-verifier zero-knowledge (HVZK), knowledge of the full challenge sequence $(c_1, \ldots, c_\mu)$ suffices to simulate an accepting transcript.

Despite their expressive power, multi-round protocols lack several structural advantages enjoyed by Σ-protocols. In particular, for Σ-protocols, both extractors and simulators can often be used as explicit algorithmic components in higher-level constructions, rather than merely as proof artifacts. For example, one only needs two accepting transcripts to run a Σ-protocol extractor, whereas one needs a much larger tree of transcripts for a multi-round proof. Similarly, for running a zero-knowledge simulator, one needs to know the complete challenge vector in advance, as opposed to knowing only one (the only) challenge in a Σ-protocol. These features hinder the use of multi-round protocols in some creative and interesting applications of Σ-protocols, including trapdoor commitments [Dam89, Fis01], proof composition, and proofs of partial knowledge [CDS94, CDM00].

Recent work has partially addressed this gap on the zero-knowledge side. The abstraction of *critical-round zero knowledge* (CRZK) [ABB+26], building on earlier notions such as k-zero knowledge [GKK+22, DG23], identifies protocols in which knowledge of a single designated challenge suffices to simulate the entire transcript. This relaxation enables simulators to be meaningfully used in cryptographic constructions, and has been shown to apply to several multi-round protocols used in practice, including discrete log-based SNARKs [GWC19, BBB+18, AF22] and the KKW framework for MPC-in-the-head with preprocessing [KKW18].

These developments raise the question of whether a similar idea can be applied for special soundness. Specifically,

Can one define a notion of special soundness for multi-round proofs that is strong enough to enable reusable extractors, analogous to those for Σ-protocols?

And moreover,

Can such a notion capture protocols already deployed in practice and unlock applications of interest?

1.1 Our Contributions

In this work, we initiate a systematic study of refined soundness notions for multi-round public-coin proof systems, giving a positive answer to the questions above. Our goal is to bridge the conceptual gap between three-move protocols and their multi-round generalizations, and to clarify when and how extractors for multi-round proofs can be made usable beyond security reductions.

Table 1. Protocols captured by the CRSS framework and the benefit of critical-round extraction over the (extended) special-soundness analysis of the underlying protocol. $\vec{h}$ is the extended tree shape, $\vec{k}$ the critical tree shape at critical round i^*_{ss}; predicate systems are omitted when trivial. The sumcheck row records a sub-protocol tree-extraction guarantee. The last column lists one illustrative benefit per protocol, not an exhaustive account. The Fischlin entries compare tree-counting bounds. Their security guarantee also requires the RUR condition of Theorem 8.

Protocol	**Extended $\vec{h}$ / critical $\vec{k}$ at i^*_{ss}**	**Illustrative Benefit**
PPR-KKW	$\vec{h} = (2, 2)$ $\vec{k} = (1, 2)$ at $i^*_{\mathsf{ss}} = 2$	Simplest instantiation of a $(1, n)$-WS scheme (Section 4) and an offline-extractable TDC (Section 5.2).
C&C-KKW	$\vec{h} = (2, 2, \boldsymbol{\Phi})$ $\vec{k} = (1, 2)$ at $i^*_{\mathsf{ss}} = 2$	Avoids the naive $\binom{m-1}{\tau-1}+1, 2)$-SS, exponential in τ; knowledge error $\kappa_{\mathrm{c\&c}}$ (Corollary 4) negligible in τ.
Σ-protocols with helper	$\vec{h} = (2, 2, \boldsymbol{\Phi})$ $\vec{k} = (1, 2)$ at $i^*_{\mathsf{ss}} = 2$	For the batched 5-move compiler, avoids the naive $\binom{m-1}{\tau-1}+1, 2)$-SS; the additive knowledge-error bound of C&C-KKW holds with n replaced by $\lvert\mathsf{Ch}\rvert$, under the conditions of Corollary 6
KZG-compiled Boolean check	$\vec{h} = (2, 2n+1)$ $\vec{k} = (1, 2n+1)$ at $i^*_{\mathsf{ss}} = 2$	Only the query round needs branching; the batching round α is critical-free. Matches the tightness goal of [KLPS26], with $\kappa_{ex} = 2(Q+1)(2n+1)/\lvert\mathbb{F}\rvert$.
Plonk	$\vec{h} = (3n+1, 3n+1, 3, 4n+21, 6)$ $\vec{k} = (1, 1, 1, 4n+21, 6)$ at $i^*_{\mathsf{ss}} = 4$	Critical extraction needs branching only in the last two challenge rounds; the first three rounds enter the extended extraction. The Fischlin application additionally requires RUR and does not apply to this variant.
Sumcheck in Spartan	$\vec{h} = \big((1, 2)^\mu, 2\big)$ $\vec{k} = \big((1, 1)^\mu, 2\big)$ at $i^*_{\mathsf{ss}} = 2\mu+1$	Deferring the μ inner Σ-protocol challenges to one final move turns μ separate per-iteration critical rounds into a single critical round at the very end.
Protogalaxy	$\vec{h} = (n, t+1, dk+1)$ $\vec{k} = (1, 1, k+1)$ at $i^*_{\mathsf{ss}} = 3$	Multi-round Fischlin bound L drops from $\Omega(n \log n \cdot dk)$ to the optimal $L = k$, independent of n, t, d; matches the $\mathcal{O}(dk)$ cost of the interactive prover.

Critical-round special soundness (CRSS). The starting point is the observation that the special soundness of multi-round protocols often requires a large tree of transcripts, but that a much smaller tree is typically sufficient to obtain a candidate witness; the rest of the tree is only needed to certify the validity of that candidate by ruling out a break of an underlying hardness assumption (e.g., binding of the commitment scheme used by the protocol). We formalize this intuition with a new notion *critical-round special soundness (CRSS).* It identifies a *critical subtree* T^* inside the extended tree of transcripts T, and requires the existence of a *critical-round extractor* CrExt that operates only on T^*, while the full tree T serves as a fall-back used to construct a reduction to the hardness assumption whenever CrExt fails. The notion comes in two flavors (Definition 14): **Plain CRSS**, capturing protocols whose extended tree carries no further structural constraint. As the simplest instantiation amenable to our applications mentioned below, we show that a partial parallel repetition of KKW (PPR-KKW) [KKW18] satisfies this notion. **Predicate CRSS**, where the critical subtree T^* and the extended tree T are equipped with predicates Φ_{cr} and $\boldsymbol{\Phi}$, respectively, that enforce structural constraints (e.g., on the challenges and on the relations among different branches of the tree). Predicate CRSS is needed to capture protocols in which the extractor for the underlying hardness assumption requires the extended tree to satisfy non-trivial structural conditions. This is the case for the cut-and-choose KKW (C&C-KKW) protocol analyzed in Section B.1. Beyond KKW, we show that CRSS captures several other multi-round protocols deployed in practice, including Σ-protocols with a trusted helper [Beu20, BGKM23], KZG-compiled polynomial IOPs and Plonk [KLPS26, LPS25], and the Protogalaxy folding scheme [EG23]. We also give a corresponding tree-extraction result for the sumcheck sub-protocol of Spartan [DG23]. Table 1 summarizes these examples together with a concrete benefit of critical-round extraction in each case.

Application 1: Witness sharing. We introduce *witness sharing* (WS), which can be seen as an amortized form of non-interactive verifiable secret sharing. In this setting, a dealer distributes a witness

w for a statement x among n recipients so that (i) each share independently certifies its validity, (ii) any $t+1$ recipients can reconstruct the witness, and (iii) any t recipients learn nothing about w.

A challenge is that the threshold required by WS from higher-level applications must be flexible relative to the tree parameters of CRSS, which are often fixed for a given instantiation of the proof system. When we instantiate with MPCitH protocols, the number n of virtual parties is typically determined by the trade-off between security and efficiency, and the threshold is often set to $t = n - 1$. Thus it is not trivial to choose them as required without considerable overhead. We give two constructions that address this challenge at different levels of generality:

- *Generic construction* (Section 4.2) from any $(1, \ldots, 1, t+1)$-CRSS proof system Π that allows to simulate $(1, \ldots, 1, t)$-subtree. This construction is intended to illustrate how the CRSS notion is concretely used in the WS framework: Use each transcript in a critical subtree as a share, and use the critical-round extractor as the reconstruction algorithm. As a result, the threshold t is directly tied to the simulatability parameter of the underlying proof system, making the interplay between CRSS and WS transparent.
- *Flexible MPCitH-based construction* (Section 4.3) that decouples the WS threshold from the MPCitH parameters, supporting arbitrary (t, n) pairs independently of the underlying MPCitH parameters, with $O(\log n)$ additional communication per virtual MPC execution. The resulting WS scheme admits hash-based instantiations and thus provides plausible post-quantum security.

We compare our approach to other potential constructions such as a simple combination of a verifiable secret sharing scheme and a non-interactive zero-knowledge proof in Section 4.4.

Application 2: Trapdoor commitments with offline trapdoor extraction. Trapdoor commitment (TDC) schemes [Dam89] allow a committer to commit to a message m and equivocate to any m' using a trapdoor. *Offline trapdoor extractability* guarantees that, given two valid and distinct openings of a commitment, the trapdoor can be extracted in strict polynomial time, without rewinding the adversary. This property is well-known for Σ-protocols [Dam89, Fis01] but no general construction was previously known from multi-round ones: the construction of [ABB+26] from CRZK only achieves *online* trapdoor extractability, which requires rewinding. We close this gap by leveraging CRSS: the critical-round extractor CrExt of a CRSS proof system runs in strict polynomial time on a small critical tree, and is precisely the offline extractor we need (Theorem 7). We further give an instantiation from a parallel-repetition variant of KKW (Section 5.2), yielding a hash-based offline-extractable TDC.

Application 3: A more efficient Multi-round Fischlin transform. The Fischlin transform [Fis05] compiles a Σ-protocol into a non-interactive argument with *straight-line* knowledge extraction in the ROM via a proof-of-work strategy. In [RT25], Rotem and Tessaro generalize this to μ-round $(k_1, \ldots, k_\mu)$-special-sound protocols, forcing the prover to probe a small challenge set in every round. To keep the prover cost polynomial, their parameter choice requires the number of rounds $\mu = O(\log \lambda / \log\log \lambda)$ for security parameter λ.

We observe that for CRSS protocols at critical round i^*_{ss}, probing is unnecessary for rounds $i < i^*_{\mathsf{ss}}$. Since the critical-round extractor only requires a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-subtree, early rounds can keep a plain Fiat-Shamir challenge. Asymptotically, only the number of rounds from i^*_{ss} needs to be bounded by $O(\log \lambda / \log\log \lambda)$.

This parameter improvement is not only of theoretical interest; it also expands the design space of efficient straight-line extractable arguments. Moreover, it reduces the prover's practical workload: Fischlin's transform forces the prover to build a transcript tree that enables extraction, while our critical-round extractor requires a smaller tree. As a result, the prover can complete the task more quickly. This is a direct practical benefit of CRSS in this application.

1.2 Technical Overview

In this part, we illustrate the core technical ideas. We first recall the limitation of the usual special soundness notion for multi-round protocols deployed in practice and describe the CRSS notion in its plain form, using KKW as a running example; we then turn to the predicate variants and to our Extract-or-Extend theorem, followed by the main applications.

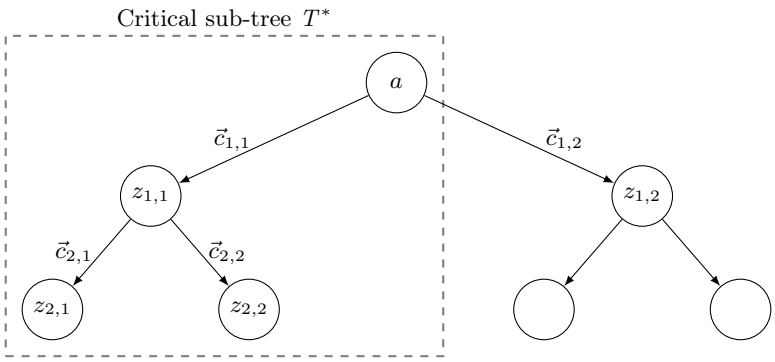

Fig. 1. Example of extended tree T and critical tree T^* for a five-move protocol (e.g., [KKW18]), satisfying $(1,2)$-CRSS at round $i^*_{\mathsf{ss}} = 2$.

From Extended Special Soundness (ExSS) to Critical-Round Special Soundness (CRSS). Recall that special soundness of a proof system for relation R enables witness extraction from an $(h_1, \ldots, h_\mu)$-tree T of accepting transcripts. As observed in the analysis of real-world protocols, however, T is often redundant when it comes to knowledge extraction for R only; a *partial* tree of transcripts T^* already contains a candidate witness for R. For instance, if the protocol has a typical commit-and-prove structure, a $(k_1, \ldots, k_\mu)$-subtree T^* of an $(h_1, \ldots, h_\mu)$-tree T may be sufficient to fully open the commitment containing a candidate witness w' for R, while the validity of w' is guaranteed only if the outer tree does *not* define two distinct openings of the commitment (i.e., a solution to the binding relation).

We formally capture the above intuition by identifying a *critical subtree* T^* of an extended tree T, as illustrated in Figure 1. A $(k_1, \ldots, k_\mu)$-tree T^* is such that $k_i \leq h_i$ for every $i \in [\mu]$ and that begins branching at some earliest round i^*_{ss}, the *critical round*. In essence, the $(k_1, \ldots, k_\mu)$-critical tree is sufficient to obtain a witness, and the outer $(h_1, \ldots, h_\mu)$-tree guarantees that the witness is indeed valid. We then define our *critical-round special soundness* (CRSS) notion by requiring the existence of both a *critical extractor* CrExt and an *extended extractor* ExExt (Definition 12). Essentially, the critical-round extractor CrExt is a refinement of ExExt that only operates on the critical subtree T^*.

To illustrate their roles, consider a $(2\mu+1)$-move public-coin argument system Π for relation R with hardness assumption R_{hd}, which is $\vec{h}$-extended special sound (Definition 3). We say that Π is (plain) CRSS if for any accepting critical subtree T^*, either:

- the critical-round extractor $\mathsf{CrExt}(T^*)$ outputs a valid witness w such that $R(x,w) = 1$, *or*
- for every extended tree $T \sqsupseteq T^*$, the extractor $\mathsf{ExExt}(T)$ outputs a solution s to the hardness assumption with public instance y, satisfying $R_{hd}(y,s) = 1$,

In other words, the critical subtree suffices for extracting a witness, while the full extended tree serves as a "fall-back option" that always yields a solution to R_{hd} whenever the critical extractor fails.

Running example: KKW. Consider the bare-bones KKW protocol for MPC-in-the-head [KKW18] (see Appendix C). In the first round, the prover commits to m independent executions of a preprocessing phase; the verifier's challenge $c_1 \in [m]$ selects the one execution that remains unopened, and the prover opens the other $m-1$. The prover then runs the online phase of an n-party MPC protocol on the unopened preprocessing material and commits to the views of the n parties; the verifier's second challenge $c_2 \in [n]$ selects one party that remains unopened, and the prover opens the views of the other $n-1$. It is easy to see the protocol is extended $(2,2)$-special sound: given two first-round challenges $c_1 \neq c'_1$ and, below c_1, two second-round challenges $c_2 \neq c'_2$, the two transcripts below c_1 together open the views of all n parties of the online execution that uses preprocessing c_1, while the branch below c'_1 opens the seed of that very preprocessing execution, since $c'_1 \neq c_1$. The extractor recomputes the preprocessing states from the seed and compares them with the states revealed by the views: if they agree, the online execution is an honest evaluation of the circuit and unmasking its input wires yields a witness; if they disagree, two consistent openings of the same commitments give a break of binding or a hash collision, i.e., a solution to R_{hd}.

Now observe that the candidate witness is computed from the $(1,2)$-subtree T^* below c_1 alone, because it opens all n views. This is the critical extractor CrExt, and $i^*_{\mathsf{ss}} = 2$ is the critical round

(Figure 1). The second branch of the extended tree is only needed as a fall-back: if the candidate is not a valid witness, then the preprocessing states behind the n views cannot be the honestly derived ones, so *any* extension of the critical tree, which necessarily opens the seed of preprocessing c_1, exhibits the inconsistency and yields a solution to R_{hd}. Hence bare-bones KKW is plain $(1,2)$-CRSS at round 2 and extended $(2,2)$-SS.

This toy example does not yet give a negligible extraction error, since its soundness error is $\max(1/m, 1/n)$; the standard remedy is parallel repetition. We encounter a subtle issue here. If all τ repetitions are run independently, two second-round challenge vectors in $[n]^\tau$ may differ in a single coordinate j, so a $(1,2)$-critical tree T^* opens all views of one repetition only, and the fall-back works only if the extended tree also opens the preprocessing of *that particular repetition*. Clearly, a mere $(2,2)$-extended tree doesn't guarantee such a condition. Our *partial parallel repetition* $\Pi_{\mathsf{PPR\text{-}KKW}}$ (Section 3.3) avoids this by repeating the first round τ times independently while sharing a single second-round challenge $c_2 \in [n]$ across all repetitions: a $(1,2)$-critical tree then opens all n views of *every* repetition, the fall-back applies to whichever repetition the extended tree opens, and the first-round term $1/m^\tau$ of the extraction error now becomes negligible. It turns out that the remaining $1/n$ term from the shared critical-round challenge will disappear in our applications described later, where the critical-round challenges are chosen by the adversary rather than sampled.

Predicate ExSS and CRSS. Protocols deployed in practice are often optimized in ways that break even extended special soundness with a small tree. For instance, the cut-and-choose optimization of KKW (C&C-KKW), used in Picnic2 [KZ20], opens $m-\tau$ of m preprocessing executions at once and runs τ online executions on the unopened ones, with $\vec{c}_1 \in [m]^\tau$ and $\vec{c}_2 \in [n]^\tau$. Two second-round challenge vectors may differ in a single coordinate j, so extraction needs an extended tree whose second first-round challenge opens the preprocessing execution c_1^j; as $\binom{m-1}{\tau-1}$ of the $\binom{m}{\tau}$ first-round challenges leave c_1^j unopened, this requires a $(\binom{m-1}{\tau-1}+1, 2)$-tree, exponential in τ. We address this by incorporating the *predicate special soundness* framework of Aardal et al. [AAB+24], only requiring extraction from trees that satisfy a system of predicates $\boldsymbol{\Phi}$ on the challenges. For C&C-KKW, one may define a predicate such as "the second first-round challenge opens some execution whose views are fully revealed in the first branch", at the price of adding to the knowledge error the fraction of challenges that violate the predicates, their *failure density*. C&C-KKW is then $(2,2,\boldsymbol{\Phi})$-predicate special sound with negligible failure densities (Appendix B). Building on this, we define *predicate CRSS*, which captures the same intuition as above with respect to a predicate Φ_{cr} on the critical subtree T^* and a predicate system $\boldsymbol{\Phi}$ on the extended tree T. The formal treatment is somewhat tedious because the predicates take subtrees as input and must be compatible with the branches inherited from the critical tree, which we detail in Sections 2.3 and 3.1.

Extract-or-Extend theorem. In our applications, an adversary does not output a single Fiat-Shamir proof but directly an accepting critical tree: $t+1$ valid shares in witness sharing, or two openings of a trapdoor commitment. Our main technical result (Theorem 2) bounds the probability that CrExt fails on such a tree by $\kappa_{ex} + \epsilon_{hd}$, where ϵ_{hd} is the probability of solving R_{hd} and

$$\kappa_{ex} = 2(Q+1) \sum_{i \neq i^*_{\mathsf{ss}}} \max\left(\tfrac{h_i - 1}{|\mathcal{C}_i|},\ \textstyle\sum_\ell p^{\mathsf{chal}}_{i,\ell} \right)$$

sums, for every round *other than the critical one*, the branching factor and the failure densities of the predicates, in the Fiat-Shamir knowledge error of predicate special soundness [AAB+24]. We prove this via the following "extract-or-extend" argument: if CrExt fails, we rewind the adversary with reprogrammed random-oracle answers, one non-critical round at a time, to complete the critical tree T^* into an extended tree $T \sqsupseteq T^*$ with $\boldsymbol{\Phi}(T) = 1$, and then $\mathsf{ExExt}(T)$ solves R_{hd}. Our analysis partially relies on the tree builders of [AFK23, AAB+24] with subtle deviations observed in Section 3.2.

Witness sharing. Consider $(1,\ldots,1,t+1)$-CRSS with critical round i^*_{ss} and a simulatability property that allows to simulate $(1,\ldots,1,t)$-subtree. Our first generic construction of WS from such a proof system is very simple: the dealer runs Π via Fiat-Shamir up to round $\mu-1$ (fixing a prefix *com*), then computes n challenges for round i^*_{ss} via a separate hash function, runs the prover n times on each challenge to generate complete transcripts $\sigma_1,\ldots,\sigma_n$, and distributes each of these as a share to the recipients. Soundness follows from the CRSS property: any $t+1$ valid shares form a critical tree from which CrExt directly extracts w. Privacy follows from the $(1,\ldots,1,t)$-simulatability of Π.

We then present a more flexible construction based on MPCitH, supporting arbitrary (t, n) pairs independently of the underlying MPCitH parameters. A naïve attempt at using (t, n)-MPC (the witness is shared among n parties using (t, n)-threshold secret sharing) in the MPCitH framework discloses the view of only one virtual player in each proof, so that no collection of t proofs recovers the witness but $t+1$ do. This causes a significant loss in knowledge soundness, since up to t simulated views can be opened. [7] Our idea is to retain the original construction with additive MPC but share the private input of each virtual player using a (t, n)-threshold secret sharing scheme. The dealer commits to the shares and opens all of them (for opened virtual players) or only one (for concealed virtual players), according to whether the virtual player's view is revealed. Directly committing to and opening n shares would incur $O(n)$ blowup in communication cost. Instead, we achieve $O(\log n)$ communication by committing to the sharing polynomial, which is generated pseudo-randomly and deterministically from a small seed, and also committing to each share using Merkle trees. When a virtual player is to be opened, the dealer opens the seed, allowing the recipients to recover all shares from the sharing polynomial and check the Merkle tree. For an unopened virtual player, the dealer opens to each recipient a distinct share along with its Merkle authentication path. This allows each recipient to verify the authenticity of the share it receives, and by collecting a quorum of such shares, the unopened virtual player's input can be recovered. Thus, parameters t and n can be set independently from the underlying MPCitH parameters with additional $O(\log n)$ communication per virtual MPC execution.

Trapdoor commitments with offline trapdoor extraction. Our starting point is a $(1, \ldots, 1, 2)$-CRSS protocol. We use a partial transcript up to the critical round as the commitment and embed a message to create a challenge for the critical round. An opening of the commitment consists of a valid response to the critical-round challenge. Since the witness allows to complete the partial transcript to a valid transcript for any critical-round challenge, the committer can equivocate by generating a different valid response to a different critical-round challenge. On the other hand, given two valid openings with different critical-round challenges, the critical-round extractor CrExt can be run on the corresponding critical tree to extract a witness, which serves as the trapdoor. Thus, the offline extractor is precisely CrExt. The CRSS property then bounds the failure probability of CrExt via the hardness assumption R_{hd} (Theorem 7).

The requirement is that the critical tree has only *two* branches at the critical round ($k_{i^*_{\mathsf{zk}}} = 2$); otherwise CrExt would need more openings than offline extraction provides. We instantiate this from PPR-KKW (Sections 3.3 and 5.2), which answers a question left open in [ABB+26]. Note that the choice of variant is critical. Since the committer chooses the critical-round challenge, it can pick the two challenges $c_{i^*_{\mathsf{zk}}} \neq c'_{i^*_{\mathsf{zk}}}$ of its openings so as to make the remaining rounds easy to answer. C&C-KKW thus does not qualify as TDC: committer who corrupts a single preprocessing execution can, whenever that execution stays unopened, produce two valid openings whose critical-round challenges $\vec{c}_2, \vec{c}'_2 \in [n]^\tau$ differ only in that execution, so that no honest execution is ever revealed together with all of its views, and offline extraction fails with probability $\binom{m-1}{m-\tau}/\binom{m}{m-\tau}$. PPR-KKW keeps the first-round challenges independent across the τ repetitions but shares a single critical-round challenge $c_2 \in [n]$, so two openings with $c_2 \neq c'_2$ reveal all n views of *every* repetition and the fall-back applies to any honest one. It is plain $(2, 2)$-extended SS and $(1, 2)$-CRSS at $i^*_{\mathsf{ss}} = 2$ (Lemma 1) and yields the first hash-based offline-extractable trapdoor commitment from a multi-round proof (Corollary 2).

Improving the Fischlin-like transform of Rotem and Tessaro. The Fischlin transform [Fis05] for three-move protocols forces the prover to perform a hash-based proof of work: it iterates over a challenge set of size $c \cdot k$ and outputs a transcript π such that $H(\pi)$ begins with ℓ leading zeros. An extractor that inspects the prover's random-oracle queries can then directly reconstruct a k-tree of accepting transcripts, without ever rewinding, yielding straight-line extractability.

Rotem and Tessaro [RT25] recently generalized Fischlin's transform to μ-round public-coin protocols that are $(k_1, \ldots, k_\mu)$-special sound. The prover runs the underlying protocol while, at each round i, probing a challenge set of size $k'_i := c \cdot k_i$: it iterates over candidate challenges via an operator $\mathsf{increment}_{\mathbf{k}'}$ that enumerates index tuples $\mathbf{j} = (j_1, \ldots, j_\mu) \in [k'_1] \times \cdots \times [k'_\mu]$ in lexicographic order, computes responses for each, and accepts the first transcript whose hash has ℓ leading zeros. [RT25] bounds the maximum number of leaves of the partial tree that a malicious prover can induce with q

[7] The adversary can make up to t consistent views, and only one view will be opened. So it has t times more chance of successful cheating.

random-oracle queries without enabling extraction. The bound takes the form of a sum over all rounds $i \in [\mu]$ of terms depending on k'_i and k_i, and parameter setting implies that $\mu = \mathcal{O}(\log \lambda / \log\log \lambda)$ if one wants to keep the prover cost polynomial.

We observe that the probing strategy of [RT25] only needs a branching of size k_i at round i *if witness extraction itself requires k_i branches at that round*. For protocols that are CRSS at critical round i^*_{ss}, the critical-round extractor CrExt rebuilds a witness from a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-critical subtree in which the early rounds do not branch. On such rounds, probing is redundant: the prover can simply derive the plain Fiat–Shamir challenge

$$c_i := H_i\big(x, a, z_1, c_1, \ldots, z_{i-1}\big).$$

Then, the straight-line extractor only rebuilds the critical subtree starting from round i^*_{ss}.

This replacement is precisely what is needed to start the summation of the bounds in [RT25] at $i = i^*_{\mathsf{ss}}$ rather than at $i = 1$. The terms corresponding to rounds $i < i^*_{\mathsf{ss}}$ no longer contribute, because at those rounds the malicious prover has a single Fiat–Shamir challenge per transcript prefix and cannot trade additional random-oracle queries for additional branches. The round-complexity constraint is consequently relaxed from $\mu = \mathcal{O}(\log \lambda / \log\log \lambda)$ to

$$\mu - i^*_{\mathsf{ss}} + 1 = \mathcal{O}(\log \lambda / \log\log \lambda),$$

i.e., only the number of rounds *from the critical round onward* must be bounded, while the number of rounds before i^*_{ss} is unrestricted. For protocols with a late critical round (such as Protogalaxy) this meaningfully reduces the prover running time induced by the transform.

1.3 Related Work

Critical Round Zero-Knowledge. In standard definitions of honest-verifier zero-knowledge for multi-round protocols, the simulator in the proof of security is required to fix in advance all verifier challenges in advance. In contrast, the recent work [ABB+26] proposes a refined notion called *critical-round special honest-verifier zero-knowledge (CRSHVZK)*, where the simulator only requires pre-fixing a single challenge corresponding to a *critical round* i^*_{zk}.

Concretely, once the simulator determines the value of $c_{i^*_{\mathsf{zk}}}$ ahead of time, it can produce a full transcript $(a, c_1, z_1, \ldots, c_{i^*_{\mathsf{zk}}}, \ldots, c_\mu, z_\mu)$ while reading the remaining challenges in real-time. More formally, the definition requires that the simulator can be decomposed into two (stateful) components, denoted SimA and SimZ. The algorithm SimA outputs the initial prover message a given the pre-selected critical challenge $c_{i^*_{\mathsf{zk}}}$. The algorithm SimZ then proceeds sequentially: upon receiving an arbitrary challenge c_i, it produces the corresponding prover response z_i for round i. [ABB+26] also introduces a black-box construction of a trapdoor commitment scheme from a multi-round proof that satisfies CRZK, albeit without the offline extractability property.

Soundness of Proof Systems. Since its introduction for Σ-protocols [Cra96], special soundness has been extended to capture a variety of public-coin proof protocols. To handle parallel soundness amplification of k-special sound protocols, [AAB+21] relaxes the extractor so it works only for transcripts that satisfy a predicate. The works in [AF22, AFR23] formalize a multi-round generalization of (possibly computational) special soundness, known as $(k_1, \ldots, k_\mu)$-special soundness. This concept is then extended to manage erroneous transcripts in several ways: almost special soundness in [BF23] provides a framework to analyze protocols based on deterministic commitments, whose opening gets checked probabilistically through a random challenge. predicate special soundness in [AAB+24] uses predicates over challenges in each round; adaptive special soundness [AKLY24] conceptualizes a useful challenge subspace with respect to a set of transcripts available in each state of the extractor; statistical special soundness in [ABO+24] allows a statistical error over all challenges in a tree of transcripts; and $\mathfrak{G}$-soundness [DFMS22] broadly describes conditions under which the extractor functions.

Secret Sharing of an NP statement. Goyal et al. [GJS19]: introduced a primitive, which they call NP secret sharing (NPSS), that transforms a statement x into several related statements $x_1, \ldots, x_n$, each including local witnesses for which a recipient can check local validity, so that enough satisfiable shares imply that x is satisfiable. NPSS is formulated differently from our witness sharing with respect to how the global validity is verified; the verification function takes the entire sharing as input, therefore

it is not necessarily clear what the local validity checks entail. Their construction follows the MPC-in-the-Head paradigm and covers the n-out-of-n case, together with a gap-threshold variant based on MPC where the privacy and recovery thresholds are separated. Applebaum and Kachlon [AK25] focus on information-theoretic privacy of NPSS, limiting themselves to 3-move IT-MPC case and covers arbitrary thresholds.

Verifiable Relation Sharing (VRS) was introduced by Applebaum et al. in [AKP22] and is more closely related to our work. There, a prover shares secret data among multiple verifiers while proving that the shared data satisfies a public relation. Protocols known as Distributed Verifier Zero-Knowledge, e.g., [ACF02,YW22,EPSW24], and Zero-knowledge over Distributed Data, e.g. [BBC+19, BGI16], can work similarly for sharing a witness. These generalize VSS and zero-knowledge over secret-shared data, and can encode verifiable sharing of an NP witness by choosing the relation appropriately. However, in these protocols verification is in general *protocol-based*, as opposed to a non-interactive local check: the verifiers run a low-round interactive protocol with broadcast/private channels and may need to agree to accept or disqualify the dealer. Besides being inherently interactive among the parties, accountability is less clean in this approach because validity is tied to the agreement among the recipients rather than a single self-contained proof artifact. We refer to Section 4.4 for further discussion and comparison. .

2 Preliminaries

Notation Throughout the paper, we use sans-serif for algorithms. Sets are denoted by calligraphic letters. Entities such as a prover, a verifier, and an adversary are denoted by capital letters. Some exceptions may be used for better readability. We use notation $x \leftarrow\!\!\$\ \mathcal{D}$ as sampling from a probability distribution $\mathcal{D}$. If $\mathcal{D}$ is a set, then we assume elements from the set are sampled uniformly. A sequence of values from 1 to n (inclusive) is denoted using $[n]$.

2.1 Public-Coin Proof System

We follow the definitions of [ACK21]. Let $R : \mathcal{X} \times \mathcal{W} \to \{0,1\}$ be a binary relation defined over instances $\mathcal{X}$ and witnesses $\mathcal{W}$. Language $\mathcal{L}_R$ associated by R is $\mathcal{L}_R := \{x \in \mathcal{X} \mid \exists w \in \mathcal{W} : R(x,w) = 1\}$. By $\mathcal{L}_{RW}$, we denote set $\mathcal{L}_{RW} := \{(x,w) \mid R(x,w) = 1\}$. $\mathcal{L}_R(\lambda)$ ($\mathcal{L}_{RW}(\lambda)$, resp.) denotes a subset of $\mathcal{L}_R$ ($\mathcal{L}_{RW}$, resp.) whose instance x is limited to λ bits. An interactive proof system Π for relation R is a pair of interactive algorithms, prover P and verifier V, where a witness w is given to P as private input and instance x is given to both P and V as common input, and V outputs $b \in \{0,1\}$ at the end of the execution. By $\langle P(w), V\rangle(x) \to b$ we denote an execution of the algorithms. A transcript of a protocol execution consists of x, b, and all content of input and output communication tapes of V. It is (perfectly) complete if, for any $(x,w) \in \mathcal{L}_{RW}$, $\langle P(w), V\rangle(x) \to 1$.

Definition 1 (Public-Coin Proof Protocol). *A $(2\mu+1)$-move public-coin proof protocol for relation R is a set of probabilistic polynomial-time algorithms A, Z, V and efficiently and uniformly sampleable space $\{\mathcal{C}_i\}_{i\in[\mu]}$ that constitutes an interactive proof system (P,V) that:*

Step 1: P runs $(st_1, a) \leftarrow \mathsf{A}(x,w)$ and sends a to V.
Step $2i$: V uniformly chooses c_i from $\mathcal{C}_i$ and sends it to P.
Step $2i+1$: P runs $(st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i)$ and sends z_i to V.
Final: V runs $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu) \to b$ and outputs b.

A transcript $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ with respect to instance x is accepting if $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu) = 1$. The protocol has μ rounds, each of them consisting of a prover message followed by a verifier challenge.

Definition 2 (Tree of Transcripts [ACK21]). *A $(k_1, \ldots, k_\mu)$-tree is a directed tree where every node at depth i $(1 \le i \le \mu)$ has exactly k_i outgoing edges. $(k_1, \ldots, k_\mu)$-tree of transcripts for a $(2\mu+1)$-move public-coin proof protocol is a set of $\prod_{i=1}^{\mu} k_i$ transcripts that can be represented by a $(k_1, \ldots, k_\mu)$-tree in the following manner. Every path of the tree corresponds to a transcript in a way that the i-th response from the prover and the i-th challenge from the verifier is assigned to the i-th node and the i-th edge from the top, respectively. Challenges assigned on a set of edges from a node must be pairwise distinct. A tree of transcripts is called accepting if every transcript in the tree is accepted.*

Below, we present definitions of soundness and zero-knowledge. Additional basic notations can be found in Appendix A.

Building on $(k_1, \ldots, k_\mu)$-Special Soundness [ACK21], the notion of *extended special soundness* (ExSS) [ABB+26] captures a broader family of protocols. For an argument system based on hardness assumptions, special soundness is defined with respect to an extended relation. Namely, given a tree T of accepted transcripts, the extended extractor either outputs a witness for the original relation R or provides a solution to a hard problem.

Definition 3 ((Extended) $\vec{k}$-Special Soundness). *Let $\mu, k_1, \ldots, k_\mu \in \mathbb{N}$ be such that $\prod_{i=1}^{\mu} k_i$ can be upper bounded by a polynomial. A $(2\mu+1)$-move public-coin argument system for relation R is extended $\vec{k}$-special sound ($\vec{k}$-ExSS for short) for hardness assumption on R_{hd} if there exists a polynomial time algorithm, ExExt, that, given an instance y and x, an $\vec{k} = (k_1, \ldots, k_\mu)$-tree of accepting transcripts T, outputs (w, s) such that either $R_{hd}(y, s) = 1$ or $R(x, w) = 1$ with probability 1. If R_{hd} is empty, we say that the protocol is $\vec{k}$-special sound ($\vec{k}$-SS for short).*

Remark 1. For the above definition to be practical, $K = \prod_{i=1}^{\mu} k_i$ must be of polynomial size, as an efficient extraction algorithm would otherwise be impossible. Accordingly, we assume a polynomial-sized K throughout our paper.

While the standard notion of a special honest verifier zero-knowledge requires having all challenges in advance, enabling simulation, the notion of CRZK [ABB+26] includes two subroutines: $(\mathsf{SimA}, \mathsf{SimZ})$, where SimA only takes the statement and a pre-fixed challenge at critical round as inputs. The formal definitions of both notions are shown below:

Definition 4 (Special Honest-Verifier Zero-Knowledge). *A $(2\mu+1)$-round public-coin proof protocol is special honest-verifier zero-knowledge if there exists a polynomial-time algorithm Sim that, for any $(x, w) \in L_{RW}$ and $c_i \in \mathcal{C}_i$ for $i \in [\mu]$, distribution of $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ generated as $(a, z_1, \ldots, z_\mu) \leftarrow \mathsf{Sim}(x, c_1, \ldots, c_\mu)$ and that of $(a', c_1, z'_1, \ldots, c_\mu, z'_\mu)$ generated as $(st_1, a') \leftarrow \mathsf{A}(x, w)$ and $(st_{i+1}, z'_i) \leftarrow \mathsf{Z}(st_i, c_i)$ for i from 1 to μ are indistinguishable.*

Definition 5 (Critical-Round (Special) Honest Verifier Zero-Knowledge [ABB+26]). *A $(2\mu+1)$-move public-coin proof protocol is critical-round special honest verifier zero-knowledge (CR-SHVZK for short) at round i^*_{zk} if there exists a set of polynomial-time algorithm $(\mathsf{SimA}, \mathsf{SimZ})$ that:*

- SimA *takes x and $c_{i^*_{\mathsf{zk}}} \in \mathcal{C}_{i^*_{\mathsf{zk}}}$ as input, and outputs (st_1, a) where st_1 is a state information and a is a prover's message at step 1.*
- SimZ *takes st_i and $c_i \in \mathcal{C}_i$, and outputs (st_{i+1}, z_i) where st_{i+1} is an updated state, and z_i is a prover's response for step $2i+1$.*
- *For any $(x, w) \in L_{RW}$ and $c_i \in \mathcal{C}_i$ for $i \in [\mu]$, distribution of $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ generated as $(st_1, a) \leftarrow \mathsf{SimA}(x, c_{i^*_{\mathsf{zk}}})$ and $(st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)$ for i from 1 to μ, and that of $(a', c_1, z'_1, \ldots, c_\mu, z'_\mu)$ generated as $(st_1, a') \leftarrow \mathsf{A}(x, w)$ and $(st_{i+1}, z'_i) \leftarrow \mathsf{Z}(st_i, c_i)$ for i from 1 to μ are indistinguishable.*

2.2 Fiat-Shamir Transformation with Open Round

Let $\Pi = (\mathsf{A}, \mathsf{Z}, \mathsf{V})$ be a $(2\mu+1)$-move public-coin proof protocol for relation R with challenge spaces $\mathcal{C}_1, \ldots, \mathcal{C}_\mu$. Throughout, we work in the random oracle model with domain-separated oracles $\mathcal{O}_i : \{0,1\}^* \to \mathcal{C}_i$ for $i \in [\mu]$, and we write $\mathcal{O} = (\mathcal{O}_1, \ldots, \mathcal{O}_\mu)$. Since every challenge of a public-coin protocol is determined by the preceding prover messages once the oracles are fixed, we hash only the prover messages.

We define the Fiat-Shamir transformation directly in the form used by our applications in Sections 4 and 5, in which the challenge of one designated round is *not* derived from the random oracle but supplied by the prover: in the trapdoor commitment of Section 5 the round-i^*_{zk} challenge encodes the committed message, and in witness sharing (Section 4) the round-μ challenges are the n share indices. We call this the Fiat-Shamir transformation *with open round i^**. The open-round challenge is hashed into all subsequent challenges, so that the remaining rounds are bound to it. The standard Fiat-Shamir transformation $\mathrm{FS}[\Pi]$ is recovered by deriving every challenge as usual, i.e., $c_i := \mathcal{O}_i(x, a, z_1, \ldots, z_{i-1})$ for all $i \in [\mu]$ and the proof is $\pi = (a, z_1, \ldots, z_\mu)$

Definition 6 (Fiat-Shamir Transformation with Open Round i^*). *Let $i^* \in [\mu]$. The Fiat-Shamir transformation of Π with open round i^*, denoted $\mathrm{FS}_{\setminus i^*}[\Pi]$, is the non-interactive proof system $(P_{\mathrm{FS}\setminus i^*}, V_{\mathrm{FS}\setminus i^*})$ defined as follows.*

- *$P^{\mathcal{O}}_{\mathrm{FS}\setminus i^*}(x, w, c_{i^*})$ takes an additional input $c_{i^*} \in \mathcal{C}_{i^*}$, runs $(st_1, a) \leftarrow \mathsf{A}(x, w)$ and, for $i = 1, \ldots, \mu$, computes*

$$c_i := \begin{cases} \mathcal{O}_i(x, a, z_1, \ldots, z_{i-1}) & (i < i^*) \\ c_{i^*} & (i = i^*) \\ \mathcal{O}_i(x, a, c_{i^*}, z_1, \ldots, z_{i-1}) & (i > i^*) \end{cases} \qquad \text{and} \qquad (st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i),$$

  and outputs the proof $\pi := (a, c_{i^*}, z_1, \ldots, z_\mu)$.
- *$V^{\mathcal{O}}_{\mathrm{FS}\setminus i^*}(x, \pi)$ checks $c_{i^*} \in \mathcal{C}_{i^*}$, recomputes the remaining challenges as above, and outputs $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.*

A transcript $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ is a transcript of $\mathrm{FS}_{\setminus i^*}[\Pi]$ *for x if c_i satisfies the above equation for every $i \in [\mu] \setminus \{i^*\}$; the challenge $c_{i^*} \in \mathcal{C}_{i^*}$ is unconstrained. A $\vec{k}$-tree of transcripts T is a* $\vec{k}$-tree of $\mathrm{FS}_{\setminus i^*}[\Pi]$-transcripts *for x if every transcript in T is a transcript of $\mathrm{FS}_{\setminus i^*}[\Pi]$ for the same x; in particular all transcripts in T share the trunk $(a, c_1, z_1, \ldots)$ up to the first branching round, and x is bound to T through the oracle inputs.*

Knowledge soundness in the random oracle model. Following [AFK23, AAB+24], a *Q-query random-oracle prover* $\mathcal{A}^{\mathcal{O}}$ is a PPT algorithm making at most Q queries to $\mathcal{O}$ and outputting a pair (x, π); it is adaptive in that x is chosen by $\mathcal{A}$. $\mathrm{FS}[\Pi]$ is *adaptively knowledge sound* with knowledge error κ if there exists a knowledge extractor which, given black-box rewinding access to any Q-query prover $\mathcal{A}$ that outputs an accepting (x, π) with probability $\epsilon > \kappa$, outputs w with $R(x, w) = 1$ with probability at least $(\epsilon - \kappa)/\mathrm{poly}(\lambda)$ in an expected polynomial number of invocations of $\mathcal{A}$.

2.3 Predicate Special Soundness

In this part, we recall the predicate special soundness framework of [AAB+24] and its adaptation introduced in [ABB+26]. We first recall the notations.

Definition 7 (Set of Partial Accepting Trees). *Let $\mu, k_1, \ldots, k_\mu \in \mathbb{N}$ and let $\Pi = (P, V)$ be a $(2\mu + 1)$-move public-coin proof for a relation R. Additionally, let $m \in [\mu]$ and $\ell \in [k_m]$. Let $\mathcal{C}_m$ be the m-th challenge set.*

- *We define $\mathbb{T}_{\mu+1}$ be the set of possible accepting transcripts for Π.*
- *We define $\mathbb{T}^{(\ell)}_{m+1}$ be the set of possible accepting $(1, \ldots, 1, \ell, k_{m+1}, \ldots, k_\mu)$-trees of transcripts for Π, and denote $\mathbb{T}_m = \mathbb{T}^{(k_m)}_{m+1}$, i.e., the set of possible accepting $(1, \ldots, 1, k_m, \ldots, k_\mu)$-trees. For each tree $t \in \mathbb{T}^{(\ell)}_{m+1}$, t can be parsed as a tuple of ℓ sub-trees $t = (t_1, \ldots, t_\ell) \in \mathbb{T}^{\ell}_{m+1}$.*
- *For $t \in \mathbb{T}_{m+1}$, we define $\mathsf{trunk}(t)$ to be the prefix $(a, c_1, z_1, \ldots, c_m, z_m)$ shared by all the transcripts in t, and $\mathsf{chal}_i(t)$ for $i \in [m]$ to be the i-th challenge $c_i \in \mathcal{C}_i$ shared by the transcripts.*
- *Let $\mathcal{C}^{(\ell)}_m$ be the set of tuples $(c_1, \ldots, c_\ell) \in \mathcal{C}_m$ with $c_{i_1} \neq c_{i_2}$ for all $i_1 \neq i_2$. These are all the combinations of m-th challenges that may occur in $\mathbb{T}^{(\ell)}_{m+1}$.*

To extend the notion of special soundness, Aardal et al. [AAB+24] introduced the notion of *predicate special soundness* (PSS), specifying granular conditions on a tree of transcripts useful for witness extraction via a system of predicates. In this work, we extend the PSS framework using the notion of *generalized challenge predicates* [ABB+26], which takes a set of subtrees and a challenge as input, and checks whether the challenge is useful for extraction with respect to the subtrees. In slightly more detail, the ℓ-th generalized challenge predicate on each level takes the first $\ell - 1$ subtrees $t_1, \ldots, t_{\ell-1}$ as input, and checks that the current challenge c_ℓ falls in a "good" domain of challenges determined by the first $\ell - 1$ subtrees. While [ABB+26] introduced generalized challenge predicates to analyze their multi-round proof of partial knowledge, we find that they are also useful for extending PSS to its critical-round counterpart.

Definition 8 (Generalized Challenge Predicates). *Let $m \in [\mu]$ and $\ell \in [k_m]$. A generalized challenge predicate on level m for the ℓ-th challenge is a deterministic function $\Phi^{\mathsf{chal}}_{m,\ell} : \mathbb{T}^{(\ell-1)}_{m+1} \times \mathcal{C}_m \to \{0, 1\}$, where the first input is assumed to be empty if $\ell = 1$. We say that a chain of level m challenge predicates is efficiently checkable if there exists a polynomial-time algorithm that, on input $(t_1, \ldots, t_{k_m}) \in \mathbb{T}_m$, checks whether there exists $\ell \in [k_m]$ such that $\Phi^{\mathsf{chal}}_{m,\ell}(t_1, \ldots, t_{\ell-1}, c_\ell) = 0$.*

In the following, we define a predicate system consisting of generalized challenge predicates [ABB+26, AAB+24], and the corresponding notion of predicate special soundness.

Definition 9 (System of Predicates). *A predicate system $\boldsymbol{\Phi}$ for a $(k_1, \ldots, k_\mu)$-tree structure is a collection of predicates for each level in the tree. The m-th level has k_m challenge predicates $\Phi^{\mathsf{chal}}_{m,1}, \ldots, \Phi^{\mathsf{chal}}_{m,k_m}$. We recursively define a series of predicate $\boldsymbol{\Phi}_m$ for $m \in [\mu+1]$, describing whether a partial tree of transcripts satisfies the predicate system. For a single accepting transcript $t \in \mathbb{T}_{\mu+1}$ we let $\boldsymbol{\Phi}_{\mu+1}(t) = 1$. For all larger subtrees $t = (t_1, \ldots, t_{k_m}) \in \mathbb{T}_m$ for some $m \in [\mu]$, then $\boldsymbol{\Phi}_m(t) = 1$ if and only if*

$$\bigwedge_{\ell \in [k_m]} \left(\boldsymbol{\Phi}_{m+1}(t_\ell) = 1 \wedge \Phi^{\mathsf{chal}}_{m,\ell}(t_1, \ldots, t_{\ell-1}, c_\ell) = 1 \right).$$

For notational convenience, we let $\boldsymbol{\Phi} = \boldsymbol{\Phi}_1$.

Definition 10 (Predicate Special Soundness). *Let $\Pi = (P, V)$ be a $2\mu+1$-message public-coin argument for a relation R. We say that Π is $(\vec{k}, \boldsymbol{\Phi})$-predicate-special-sound for $\vec{k} = (k_1, \ldots, k_\mu)$ and a predicate system $\boldsymbol{\Phi}$ if there exists a polynomial time algorithm which given a statement x and a $\vec{k}$-tree of accepting transcripts t for this statement with $\boldsymbol{\Phi}(t) = 1$ always outputs a witness w such that $R(x, w) = 1$.*

Following [AAB+24, ABB+26], we next define the failure density of a generalized challenge predicate, which upper bounds the fraction of bad challenges for any fixed set of good subtrees. Essentially, each failure density contributes to the knowledge error, and thus we need to ensure that the failure density is sufficiently small for the overall knowledge error to be negligible. Our formulation slightly deviates from [ABB+26, Def. 12]: there, a tuple of subtrees $(t_1, \ldots, t_{\ell-1})$ is "good" as soon as the level-m challenge predicates $\Phi^{\mathsf{chal}}_{m,1}, \ldots, \Phi^{\mathsf{chal}}_{m,\ell-1}$ are true, while the subtrees themselves are arbitrary accepting trees. We additionally require every prefix subtree t_i to satisfy the lower-level predicate system, $\boldsymbol{\Phi}_{m+1}(t_i) = 1$. This is a strictly weaker requirement on the predicate (fewer tuples must be handled), and lets a level-m density exploit structure that lower-level predicates already guarantee on the prefix subtrees; Lemma 4 is an example where this is essential. The change does not impact the knowledge error, since the extractor behind Theorem 1 only ever queries a density on tuples of this restricted form (Remark 2).

Definition 11 (Failure Density of Generalized Challenge Predicates). *Let $m \in [\mu]$ and $\ell \in [k_m]$. Define a set of good subtrees*

$$\mathsf{GoodTree}_{m,\ell-1} = \left\{ (t_1, \ldots, t_{\ell-1}) \in \mathbb{T}^{(\ell-1)}_{m+1} \;\middle|\; \forall i \in [\ell-1] : \boldsymbol{\Phi}_{m+1}(t_i) = 1 \wedge \Phi^{\mathsf{chal}}_{m,i}(t_1, \ldots, t_{i-1}, c_i) = 1 \right\}$$

using the shorthand $c_i = \mathsf{chal}_m(t_i)$. That is, a tuple of subtrees is good if it satisfies the predicate system so far: each prefix subtree t_i is internally valid with respect to the lower-level predicates, and the first $\ell-1$ level-m challenge predicates are true (cf. the failure density of commitment predicates in [AAB+24, Def. 5.6]). For fixed $\ell-1$ subtrees in $\mathbb{T}^{(\ell-1)}_{m+1}$, consider the set of bad ℓ-th challenges such that $\Phi^{\mathsf{chal}}_{m,\ell}$ fails,

$$\mathsf{BadChal}_{m,\ell}(t_1, \ldots, t_{\ell-1}) = \left\{ c \in \mathcal{C}_m \setminus \{c_1, \ldots, c_{\ell-1}\} \;\middle|\; \Phi^{\mathsf{chal}}_{m,\ell}(t_1, \ldots, t_{\ell-1}, c) = 0 \right\}.$$

The challenge predicate $\Phi^{\mathsf{chal}}_{m,\ell}$ has failure density $p^{\mathsf{chal}}_{m,\ell}$ if it always holds that $|\mathsf{BadChal}_{m,\ell}(t_1, \ldots, t_{\ell-1})| \leq p^{\mathsf{chal}}_{m,\ell} |\mathcal{C}_m|$ for all $(t_1, \ldots, t_{\ell-1}) \in \mathsf{GoodTree}_{m,\ell-1}$.

We restate Theorem 5.1 of [AAB+24] adapted by [ABB+26], associating PSS and failure densities with Fiat-Shamir knowledge soundness.

Theorem 1 ([AAB+24, ABB+26]). *Let $\Pi = (P, V)$ be a $(\vec{k}, \boldsymbol{\Phi})$-predicate-special-sound argument for a relation R with the corresponding failure density $\{p^{\mathsf{chal}}_{m,\ell}\}_{m \in [\mu], \ell \in [k_m]}$. If a chain of level m challenge predicates $\Phi^{\mathsf{chal}}_{m,1}, \ldots, \Phi^{\mathsf{chal}}_{m,k_m}$ is efficiently checkable for all $m \in [\mu]$, then the Fiat-Shamir transformation $\mathrm{FS}[\Pi]$ is adaptively knowledge sound for the relation R with knowledge error*

$$2(Q+1) \sum_{i=1}^{\mu} \max \left(\frac{k_i - 1}{|\mathcal{C}_i|}, \sum_{\ell=1}^{k_i} p^{\mathsf{chal}}_{i,\ell} \right),$$

where Q is the number of random oracle queries made by the prover. The number of times that the knowledge extractor invokes the prover is in expectation at most $K + Q(K-1)$, where $K = \prod_{i=1}^{\mu} k_i$.

Remark 2 (Failure densities are only invoked on good tuples). The knowledge extractor underlying Theorem 1 [AAB+24, Appendix H] is the extractor of Attema et al. [AFK23] that builds the tree of transcripts depth-first, with the sole modification that it aborts as soon as any predicate evaluates to 0. Consequently, at the moment the ℓ-th sibling on level m is sampled, the subtrees $t_1, \ldots, t_{\ell-1}$ already in hand satisfy $\boldsymbol{\Phi}_{m+1}(t_i) = 1$ and $\Phi^{\mathsf{chal}}_{m,i}(t_1, \ldots, t_{i-1}, c_i) = 1$ for all $i \in [\ell - 1]$, i.e., $(t_1, \ldots, t_{\ell-1}) \in \mathsf{GoodTree}_{m,\ell-1}$ as in Definition 11. The failure density $p^{\mathsf{chal}}_{m,\ell}$ is therefore only ever applied to tuples in $\mathsf{GoodTree}_{m,\ell-1}$, and the knowledge error analysis of [AAB+24, ABB+26] goes through unchanged with the condition $\boldsymbol{\Phi}_{m+1}(t_i) = 1$ included in $\mathsf{GoodTree}_{m,\ell-1}$.

3 Critical-Round Special Soundness

3.1 Definition of Critical-Round Special Soundness

This section introduces the CRSS notion formally. See Section 1.2 for intuition. We begin by reviewing the notion of predicate special soundness (see Section 2.3) explicitly for extended relations. This is just for notational convenience.

Definition 12 (Predicate SS for extended relation). *Let $\Pi = (P, V)$ be a $2\mu+1$-message public-coin argument for a relation R. We say that Π is $(\vec{h}, \boldsymbol{\Phi})$-predicate-special-sound for $\vec{h} = (h_1, \ldots, h_\mu)$ and a predicate system $\boldsymbol{\Phi}$ if there exists a polynomial time algorithm ExExt which given a statement x, y and a $\vec{h}$-tree of accepting transcripts T for this statement with $\boldsymbol{\Phi}(T) = 1$ always outputs a witness pair (w, s) such that $R(x, w) = 1$ or $R_{hd}(y, s) = 1$ holds.*

Next, we define the predicate critical-round special soundness (CRSS), which argues that either the critical extractor CrExt extracts a witness from a critical subtree, or the extended extractor ExExt breaks hardness assumptions from any extended tree containing the critical subtree. Before defining CRSS, we first formalize relation between the two trees.

Definition 13 (Tree Containment). *Let $\vec{k}, \vec{h} \in \mathbb{N}^\mu$ with $k_i \le h_i$ for all $i \in [\mu]$. Let T^* and T be ordered $\vec{k}$- and $\vec{h}$-trees of transcripts for the same statement, respectively. We say $T \sqsupseteq T^*$, if they have the same trunk and, recursively, the ℓ-th subtree of T contains the ℓ-th subtree of T^* for every $\ell \in [k_m]$ at each level $m \in [\mu]$. At level $\mu + 1$, containment means equality of transcripts.*

Definition 14 (Predicate/Plain CRSS). *Let $\vec{h}$ and $\vec{k}$ be vectors in $\mathbb{N}^\mu$. Let Π be a $(2\mu+1)$-move public-coin extended $(\vec{h}, \boldsymbol{\Phi})$-special sound argument system for a relation R with hardness assumption on R_{hd} and extractor ExExt. Π is said to be critical-round $(\vec{k}, \Phi_{cr})$-special sound (Predicate CRSS for short) at round $i^*_{\mathsf{ss}} \in [\mu + 1]$ if:*

- *for all $i \in [\mu]$, $h_i \ge k_i$; for all $i < i^*_{\mathsf{ss}}$, $k_i = 1$; and $k_{i^*_{\mathsf{ss}}} \ge 2$ if $i^*_{\mathsf{ss}} \le \mu$ (the case $i^*_{\mathsf{ss}} = \mu + 1$ means $k_i = 1$ for all $i \in [\mu]$, i.e., the critical tree is a single transcript);*
- *there exist a polynomial time algorithm CrExt such that, for any $\vec{k}$-tree of accepting transcripts T^* satisfying $\Phi_{cr}(T^*) = 1$, either $\mathsf{CrExt}(T^*)$ outputs w satisfying $R(x, w) = 1$, or for any extended $\vec{h}$-tree of transcripts $T \sqsupseteq T^*$ such that $\boldsymbol{\Phi}(T) = 1$, $\mathsf{ExExt}(T)$ outputs s satisfying $R_{hd}(y, s) = 1$.*

If both predicates $(\boldsymbol{\Phi}, \Phi_{cr})$ are trivial (empty), we say Π is plain *CRSS.*

3.2 Extract-or-Extend Theorem for CRSS Proof Systems

In the following sections, we will essentially require that any adversarially crafted critical tree T^* be useful for extracting a witness w; if T^* does not yield a valid witness, one should be able to solve the hard relation R_{hd} by rewinding the adversary to complete T^* into an extended tree $T \sqsupseteq T^*$. We encounter two subtleties here. First, the extended extractor only helps if $\boldsymbol{\Phi}(T) = 1$, so the branches of T^* contained in T must already be good for $\boldsymbol{\Phi}$; we capture this by requiring that Φ_{cr} implies the restriction $\boldsymbol{\Phi}|_{\vec{k}}$ of $\boldsymbol{\Phi}$ to the critical tree. Second, in both the WS and TDC applications *the challenges in round i^*_{ss} are chosen by the adversary* rather than derived from the random oracle (see

the transformation $\mathrm{FS}_{\setminus i^*_{\mathsf{ss}}}[\Pi]$ (Definition 6). We can still enable the reduction if $k_{i^*_{\mathsf{ss}}} = h_{i^*_{\mathsf{ss}}}$ and neither predicate inspects round-i^*_{ss} challenges, so the tree builder never needs to rewind that round and its contribution to the knowledge error disappears. To formalize these intuitions we first introduce the restriction of a predicate system and then state our main "extract-or-extend" theorem (Theorem 2).

Definition 15 (Restriction of a Predicate System). *Let $\vec{k}, \vec{h} \in \mathbb{N}^\mu$ with $\vec{k} \le \vec{h}$, and let $\boldsymbol{\Phi}$ be a predicate system for a $\vec{h}$-tree structure. The restriction $\boldsymbol{\Phi}|_{\vec{k}}$ of $\boldsymbol{\Phi}$ to $\vec{k}$-trees is the predicate $(\boldsymbol{\Phi}|_{\vec{k}})_1$, where $(\boldsymbol{\Phi}|_{\vec{k}})_m$ for $m \in [\mu+1]$ is defined recursively as follows. For a single accepting transcript $t^* \in \mathbb{T}_{\mu+1}$, $(\boldsymbol{\Phi}|_{\vec{k}})_{\mu+1}(t^*) = 1$. For a level-$m$ subtree $t^* = (t^*_1, \ldots, t^*_{k_m})$ of a $\vec{k}$-tree, with $m \in [\mu]$, $(\boldsymbol{\Phi}|_{\vec{k}})_m(t^*) = 1$ if and only if both of the following hold:*

(a) for every $\ell \in [k_m]$, $(\boldsymbol{\Phi}|_{\vec{k}})_{m+1}(t^*_\ell) = 1$;

(b) for every $\ell \in [k_m]$ and every tuple $(t_1, \ldots, t_{\ell-1}) \in \mathbb{T}^{(\ell-1)}_{m+1}$ such that $t_j \sqsupseteq t^*_j$ and $\boldsymbol{\Phi}_{m+1}(t_j) = 1$ for all $j < \ell$, it holds that

$$\Phi^{\mathsf{chal}}_{m,\ell}(t_1, \ldots, t_{\ell-1}, c_\ell) = 1, \qquad \text{where } c_\ell = \mathsf{chal}_m(t^*_\ell).$$

For $\ell = 1$ the tuple is empty and the condition reads $\Phi^{\mathsf{chal}}_{m,1}(c_1) = 1$.

Here $t_j \sqsupseteq t^*_j$ is the containment of Definition 13 applied to subtrees.

Intuitively, $\boldsymbol{\Phi}|_{\vec{k}}$ retains from $\boldsymbol{\Phi}$ the constraints on the branches of a $\vec{k}$-tree that are preserved under Definition 13. Since generalized challenge predicates take subtrees as input, every rightmost challenge must remain "good" whenever its preceding sibling subtrees are completed to extended ones This is imposed by condition (b), essentially requiring the ℓ-th critical challenge c_ℓ to satisfy $\Phi^{\mathsf{chal}}_{m,\ell}$ with respect to *any* $\boldsymbol{\Phi}$-compliant extension of the preceding critical subtrees $t^*_1, \ldots, t^*_{\ell-1}$, since it is the tree builder, not the adversary, that decides how they are completed. Note that $\boldsymbol{\Phi}|_{\vec{k}}(T^*) = 1$ only constrains the preserved branches of T^*: it neither asserts that an accepting extension $T \sqsupseteq T^*$ exists nor that the additional branches of such an extension satisfy $\boldsymbol{\Phi}$.

Theorem 2 (Extract-or-Extend). *Let $\vec{h} \in \mathbb{N}^\mu$ with $h_i \le |\mathcal{C}_i|$ for all $i \in [\mu]$ and $\prod_{i=1}^{\mu} h_i$ polynomially bounded. Let Π be an extended $(\vec{h}, \boldsymbol{\Phi})$-SS and $(\vec{k}, \Phi_{cr})$-CRSS argument system for relation R with hardness assumption on R_{hd}, with critical round $i^* = i^*_{\mathsf{ss}} \in [\mu]$ and $\vec{k} = (1, \ldots, 1, k_{i^*}, 1, \ldots, 1)$ where $k_{i^*} = h_{i^*}$, i.e., the critical tree branches only at the critical round, and there it branches as much as the extended tree. Let $\mathsf{CrExt}, \mathsf{ExExt}$ be fixed deterministic extractors of Definition 14. Consider the Fiat-Shamir transformation $\mathrm{FS}_{\setminus i^*}[\Pi]$ with open round i^* (Definition 6), where $\mathcal{O} = (\mathcal{O}_1, \ldots, \mathcal{O}_\mu)$ are the domain-separated random oracles of Section 2.2. Let $\mathcal{A}^{\mathcal{O}}(y)$ be any PPT adversary outputting a statement x together with a $\vec{k}$-tree of $\mathrm{FS}_{\setminus i^*}[\Pi]$-transcripts T^* for x (Definition 6), i.e., with the critical-round challenges chosen by $\mathcal{A}$ and all other challenges derived from $\mathcal{O}$, or $\perp$. W.l.o.g. $\mathcal{A}$ queries $\mathcal{O}$ on every oracle input of the transcripts it outputs. Assume that, for every accepting tree T^* of $\mathrm{FS}_{\setminus i^*}[\Pi]$-transcripts,*

$$\Phi_{cr}(T^*) = 1 \quad \Longrightarrow \quad \boldsymbol{\Phi}|_{\vec{k}}(T^*) = 1. \tag{1}$$

Then it holds that

$$\Pr\left[\begin{array}{l|l} T^* \text{ is accepting} & y \leftarrow \mathsf{Gen}(1^\lambda) \\ \wedge\ \Phi_{cr}(T^*) = 1 & (x, T^*) \leftarrow \mathcal{A}^{\mathcal{O}}(y) \\ \wedge\ R(x, w) = 0 & w \leftarrow \mathsf{CrExt}(x, T^*) \end{array}\right] \le \kappa_{ex} + \epsilon_{hd}\theta,$$

where

$$\kappa_{ex} = 2(Q+1) \sum_{i \neq i^*} \max\left(\tfrac{h_i - 1}{|\mathcal{C}_i|}, \sum_{\ell=1}^{h_i} p^{\mathsf{chal}}_{i,\ell}\right), \quad \theta = \prod_{i \neq i^*} \left(1 - \tfrac{h_i - 1}{|\mathcal{C}_i|}\right).$$

Here Q denotes the number of queries of $\mathcal{A}$ to $\mathcal{O}$, Gen is the instance generator of the hardness assumption R_{hd}, and ϵ_{hd} denotes the maximum probability, over $y \leftarrow \mathsf{Gen}(1^\lambda)$, that an algorithm running in expected time polynomial in the running time of $\mathcal{A}$ and in Q outputs s with $R_{hd}(y, s) = 1$.

Proof. We first outline our proof. Our goal is to build, from any critical tree T^* on which CrExt fails, an extended tree $T \sqsupseteq T^*$ with $\boldsymbol{\Phi}(T) = 1$ by rewinding the adversary, so that the extended extractor ExExt of Definition 14 outputs a solution s to the hard instance y The hardness of R_{hd} then bounds the probability that CrExt fails. The proof follows the tree builder of Attema, Fehr and Klooß [AFK23] and its extension with predicates by Aardal et al. [AAB+24] As usual, the adversary is rerun with reprogrammed oracle answers, one level at a time, until $h_m - 1$ accepting siblings are found at each level, and the losses are analyzed within the abstract sampling game. Our proof deviates from these works as follows:

- The adversary outputs a $\vec{k}$-tree T^* instead of a single transcript, and the builder must complete T^* into an extended tree $T \sqsupseteq T^*$.
- The challenges of the critical round i^* are chosen by the adversary itself ($\mathrm{FS}_{\setminus i^*}[\Pi]$, Definition 6), so the builder cannot reprogram the oracle at that round. Since $k_{i^*} = h_{i^*}$, it does not need to, and round i^* contributes neither to the knowledge error nor to the running time.
- The generalized challenge predicates of Definition 8 take completed sibling subtrees as input and therefore the outer predicate $\boldsymbol{\Phi}$ must at least accept an existing critical tree T^*. This is handled by the compatibility condition (1) through the restriction $\boldsymbol{\Phi}|_{\vec{k}}$ of Definition 15, whereas the sampled branches are handled by the failure densities of Definition 11 as in [AAB+24].

Let $\mathcal{A}'$ run $\mathcal{A}$ and output T^* only if it is accepting, its sampled challenges are consistent with the random oracle, $\Phi_{cr}(T^*) = 1$ and $R(x, \mathsf{CrExt}(x, T^*)) = 0$, and output $\perp$ otherwise. Let δ be its success probability over $y \leftarrow \mathsf{Gen}(1^\lambda)$, the random oracle and the coins of $\mathcal{A}$, which corresponds to the probability on the left-hand side of the inequality in the theorem. Let

$$N_m = |\mathcal{C}_m|, \qquad \alpha_m = \max\left(\frac{h_m - 1}{N_m}, \sum_{\ell=1}^{h_m} p^{\mathsf{chal}}_{m,\ell}\right), \qquad S = \sum_{m \in [\mu] \setminus \{i^*\}} \alpha_m.$$

Fix the hard instance y and the coins of $\mathcal{A}$. All bounds below are conditional on this choice, and since the loss terms do not depend on it, they continue to hold after averaging over y and the coins of $\mathcal{A}$. We therefore keep writing δ for the (averaged) success probability of $\mathcal{A}'$. Recall that $k_m = 1$ for all $m \neq i^*$.

We construct a tree builder TB given the input as T^* and access to $\mathcal{A}'$, which attempts to build an extended $\vec{h}$-tree $T \sqsupseteq T^*$. After an initial run of $\mathcal{A}'$, TB completes each recorded subtree recursively. At each round $m \neq i^*$, it first completes the recorded child, whose challenge is c with oracle input q. It then samples challenges without replacement from $\mathcal{C}_m \setminus \{c\}$ and reruns $\mathcal{A}'$ with only the answer at q changed. It preserves the child at input q if its recursive completion succeeds, until it obtains h_m children. It fails if the first completion fails, the challenge set is exhausted, or a challenge predicate fails. Completed children are kept in sampling order. Predicate checks may be deferred until the whole chain is complete. Reprogramming within each recursive call is restored on return.

At round i^* of the main claim, TB only completes the $k_{i^*} = h_{i^*}$ supplied children, without sampling challenges. Condition (1) guarantees the predicates on preserved branches after the preceding siblings are completed. Thus, successful completion gives an accepting tree $T \sqsupseteq T^*$ with $\boldsymbol{\Phi}(T) = 1$.

We analyze TB using the sampling argument of [AFK23, AAB+24]. View the oracle as a random table J. Fix the random tapes for lower-level completions independently of J, reset them at each call, and use independent randomness for sampling at the current level. For a round-m input q, fix all entries except $J(q)$. Let G_q be the challenges for which $\mathcal{A}'$ outputs a tree containing q whose recorded child can be completed, and let $a_q = |G_q|$. Then a_q is independent of $J(q)$ and the probability of such an initial output is a_q/N_m. If the initial run does not query q, changing $J(q)$ cannot change its output, so $a_q = 0$. Hence

$$\sum_q \Pr[a_q > 0] \leq Q. \tag{2}$$

For $h_m \geq 2$, inputs with $0 < a_q < 2(h_m - 1)$ contribute at most $2(h_m - 1)/N_m$ each to the failure probability. For $a_q \geq 2(h_m - 1)$, the next successful challenge at position ℓ is uniform among the $a_q - \ell + 1$ remaining elements of G_q. By Definition 11, its predicate fails for at most $p^{\mathsf{chal}}_{m,\ell} N_m$ challenges. Thus the probability of an initial success followed by a predicate failure is at most

$$\frac{a_q}{N_m} \sum_{\ell=2}^{h_m} \frac{p^{\mathsf{chal}}_{m,\ell} N_m}{a_q - \ell + 1} \leq 2 \sum_{\ell=2}^{h_m} p^{\mathsf{chal}}_{m,\ell}.$$

Here $a_q \geq h_m$, so the challenge set is not exhausted. For $h_m = 1$, there is no additional failure. Taking an average and using (2) bounds the total loss at round m by $2Q\alpha_m$.

Let p_m be the probability that the initial output exists and all its preserved level-m nodes are completed, with $p_{\mu+1} = \delta$. By a union bound over q, we obtain $p_m \geq p_{m+1} - 2Q\alpha_m$ for all $m \neq i^*$. At the unsampled round, compatibility gives $p_{i^*} = p_{i^*+1}$. Therefore,

$$\Pr[\boldsymbol{\Phi}(T) = 1 \ \wedge \ T \sqsupseteq T^*] \geq \delta - 2QS. \tag{3}$$

To bound the running time, we first show that, for fixed q, each challenge is visited as a further sample with probability at most

$$\frac{h_m - 1}{N_m}, \tag{4}$$

averaged over the initial value $J(q)$. If $a_q \geq h_m$, this probability equals $(h_m - 1)/N_m$ by symmetry, both for challenges in G_q and outside G_q. If $a_q < h_m$, it is at most $a_q/N_m \leq (h_m - 1)/N_m$. There are no further samples when $a_q = 0$ or $h_m = 1$. Summing over challenges and then over q bounds the expected number of further calls to $\mathcal{A}'$ by $(h_m - 1)Q$. Let C_m be the expected calls to $\mathcal{A}'$ for completing all preserved level-m nodes, including the initial call once. Let $\gamma_q(J)$ be the further calls for completing the recorded child at q, and be zero if no such node exists. Since $k_m = 1$, $\mathbb{E}[\sum_q \gamma_q(J)] = C_{m+1} - 1$. (4) holds for each fixed candidate and fixed lower-level tapes. Multiplying it by the completion cost for that candidate bounds the additional child completion cost by $(h_m - 1)(C_{m+1} - 1)$. Consequently,

$$C_m \leq h_m C_{m+1} + (h_m - 1)Q \quad (m \neq i^*), \qquad C_{\mu+1} = 1.$$

At the unsampled round, $C_{i^*} = C_{i^*+1}$, giving $C_1 \leq H + Q(H - 1)$ for $H = \prod_{m \neq i^*} h_m$. Rejecting repeated challenges takes at most $(r + 1)(1 + \log N_m)$ expected samples for r further distinct values, independently of their order. Together with polynomial tree size and efficient predicate checks, the same cost argument gives expected polynomial running time.

By definition 14 of Predicate CRSS, successful completion yields $(w', s) \leftarrow \mathsf{ExExt}(x, y, T)$ with $R_{hd}(y, s) = 1$. Hence the reduction $\mathsf{ExExt} \circ \mathsf{TB}$ solves R_{hd} with probability at least $\delta - 2QS$ over $y \leftarrow \mathsf{Gen}(1^\lambda)$, so the hardness assumption on R_{hd} gives $\delta - 2QS \leq \epsilon_{hd}$. Since $\kappa_{ex} = 2(Q + 1)S$, $0 < \theta \leq 1$, $1 - \theta \leq S$ and $\delta \leq 1$,

$$\theta(\delta - 2QS) - (\delta - \kappa_{ex}) = 2S - (1 - \theta)(\delta - 2QS) \geq 0.$$

Finally, we obtain $(\delta - \kappa_{ex})/\theta \leq \epsilon_{hd}$, and hence $\delta \leq \kappa_{ex} + \epsilon_{hd}\theta$. Thus $\theta \leq 1$ is the factor relating the success probability of the tree builder, which completes T^* into an extended $\vec{h}$-tree by rewinding $\mathcal{A}'$ on the rounds other than i^*, to that of $\mathcal{A}'$. The tree builder succeeds with probability at least $(\delta - \kappa_{ex})/\theta$.

The argument also applies when $\mathcal{A}$ chooses x, since the oracle input fixes the statement.

□

3.3 Instantiations of Critical-Round Proof Systems

We now analyze the CRSS property of several proof systems in practice. We start with the simplest and useful example for our applications: a variant of the KKW protocol for MPC-in-the-head [KKW18] that satisfies plain CRSS, i.e., without any predicate. It will serve as the running instantiation of the witness sharing schemes of Section 4 and of the offline trapdoor-extractable commitments of Section 5. We then summarize other protocols we analyze, whose details are deferred to Appendix B.

PPR-KKW: KKW with partial parallel repetition. Recall the bare-bones 5-round KKW protocol of Appendix C: the prover commits to m independent executions of the preprocessing phase, the verifier asks to open all but one of them, the prover runs the online MPC protocol with n parties using the unopened preprocessed randomness, and the verifier asks to open the views of all but one party. We consider the variant *partial parallel repetition KKW* (PPR-KKW) and denote it by $\Pi_{\mathsf{PPR\text{-}KKW}}$, in which the first round is repeated τ times independently while the second-round challenge is shared across all repetitions.

Protocol $\Pi_{\mathsf{PPR\text{-}KKW}}$. For each repetition $j \in [\tau]$, the prover and verifier interact as follows:

1. Prover generates m independent preprocessing materials and commits to them.

2. Verifier asks prover to open $m-1$ of the preprocessing materials, leaving one specified by challenge $c_1^j \in [m]$ unopened.
3. Prover uses the unopened preprocessing material to execute the MPC protocol and sends the masked input wires and commitments to the messages sent by n parties during the execution.
4. Verifier asks prover to open all but one party's view in the simulation of the MPC protocol, leaving one specified by challenge $c_2 \in [n]$ unopened.
5. Prover opens the $n-1$ views by opening the corresponding preprocessing material and messages that can be used to verify the correctness of the MPC execution.

Notice that $\Pi_{\mathsf{PPR\text{-}KKW}}$ is a 5-move protocol with challenge spaces $\mathcal{C}_1 = [m]^\tau$ and $\mathcal{C}_2 = [n]$: the first-round challenges $\vec{c}_1 = (c_1^1, \ldots, c_1^\tau)$ are sampled independently for each repetition, while the single second-round challenge c_2 is shared by all repetitions.

Lemma 1. *Assume that the hash function used is collision-resistant and that the commitment scheme used is computationally binding. Then $\Pi_{\mathsf{PPR\text{-}KKW}}$ is extended $(2,2)$-special sound with trivial $\boldsymbol{\Phi}$, and plain $(1,2)$-CRSS at round $i^*_{\mathsf{ss}} = 2$. Moreover, $\Pi_{\mathsf{PPR\text{-}KKW}}$ is CRSHVZK at round $i^*_{\mathsf{zk}} = 2$.*

Proof. **CRSHVZK.** CRSHVZK at round $i^*_{\mathsf{zk}} = 2$ immediately follows from the corresponding property of bare-bones KKW [ABB+26, Theorem 7], where the authors provide a simulator that takes $c_2 \in [n]$ as input and completes the remaining transcript sequentially for a single repetition. We obtain the corresponding CRSHVZK simulator for $\Pi_{\mathsf{PPR\text{-}KKW}}$ by independently running τ independent instances their simulator on the same instance and challenge c_2.

Extended $(2,2)$-special soundness. Consider an accepting $(2,2)$-tree T with first-round challenges $\vec{c}_1 \neq \vec{c}_1'$ and, below $\vec{c}_1$, two second-round challenges $c_2 \neq c_2'$. Since c_2 is shared across repetitions, the two transcripts below $\vec{c}_1$ together open the views of all n parties in *every* repetition $j \in [\tau]$. Since $\vec{c}_1 \neq \vec{c}_1'$, there exists a repetition $j \in [\tau]$ such that $c_1^j \neq c_1'^j$. Thus, the preprocessing execution c_1^j, which is unopened below $\vec{c}_1$, is opened below $\vec{c}_1'$. For this j, ExExt recomputes all n preprocessing states from the opened seed and compares them with the states revealed by the n views. If they differ, both are consistent with the same commitments and hashes verified in the accepting transcripts, and ExExt outputs a collision of the hash function or a break of the binding property of the commitment scheme. Otherwise, repetition j is an honest MPC evaluation of the circuit under correct preprocessing, and ExExt unmasks the input wires with the input masks contained in the states to obtain a witness w. Since the transcripts are accepting, the evaluation outputs 1 and $R(x,w) = 1$.

Plain $(1,2)$-CRSS at $i^*_{\mathsf{ss}} = 2$. Given a $(1,2)$-tree T^*, i.e., two accepting transcripts with the same $\vec{c}_1$ and $c_2 \neq c_2'$, the critical extractor CrExt reconstructs, for each repetition $j \in [\tau]$, a candidate witness w_j from the n opened views as above and outputs the first w_j with $R(x,w_j) = 1$, if any. Suppose CrExt fails, and let $T \sqsupseteq T^*$ be any extended $(2,2)$-tree. Its second branch has a first-round challenge $\vec{c}_1' \neq \vec{c}_1$. Let $j \in [\tau]$ a repetition whose preprocessing seed is completely opened. If the states recomputed from that seed agreed with the states revealed by the views of repetition j in T^*, the argument above would give $R(x,w_j) = 1$, contradicting the assumption that CrExt fails. If these states differ, on the other hand, $\mathsf{ExExt}(T)$ outputs a hash collision or a binding break. This is the fall-back clause of Definition 14, with trivial $\boldsymbol{\Phi}$ and Φ_{cr}. □

Note that the Fiat-Shamir transform of $\Pi_{\mathsf{PPR\text{-}KKW}}$ has knowledge error $2(Q+1)(1/m^\tau + 1/n)$ by Theorem 1, which is on its own useless due to the non-negligible term. This is exactly the situation the Extract-or-Extend theorem (Theorem 2) is designed for: in the applications of Sections 4 and 5, round 2 is the critical round whose challenges are chosen by the adversary, and Theorem 2 excludes the non-negligible second-round term $1/n$ from the extraction error. Looking ahead, $\Pi_{\mathsf{PPR\text{-}KKW}}$ yields a $(1,n)$-witness sharing scheme (Section 4) and an offline trapdoor-extractable commitment with message space $[n]$ (Section 5.2).

Further instantiations. Table 1 summarizes the further protocols we analyze in Appendix B, together with the concrete benefit of critical-round extraction in each case. The cut-and-choose optimization of KKW (C&C-KKW, used in Picnic2 [KZ20]) and the batched Σ-protocols with helper of [Beu20] require the predicate variant of CRSS; there, critical-round extraction avoids a special-soundness parameter exponential in the cut-and-choose repetition count (Lemma 3 and Corollary 5). The KZG-compiled Boolean check and the witness-revealing Protogalaxy protocol are plain CRSS with a critical tree that branches only from a single late round (Lemmas 7 and 14), which reduces

Table 2. Which protocols meet the hypotheses of the Extract-or-Extend theorem and of the applications. μ is the number of rounds, and i^*_{ss} and i^*_{zk} are the critical rounds for special soundness and for zero knowledge; the next two columns are the shape conditions of Theorem 2 on the critical tree, namely $k_i = 1$ for all $i \neq i^*_{\mathsf{ss}}$ and $k_{i^*_{\mathsf{ss}}} = h_{i^*_{\mathsf{ss}}}$; "Pred." lists the non-trivial predicate systems. Fischlin additionally requires random-challenge unique responses as in [RT25, Definition 5]. Witness sharing has two routes: the generic construction of Section 4.2 and the MPCitH-based construction of Section 4.3. Application entries refer to negligible-error guarantees, with offline extraction for TDC. The C&C-KKW, helper and Protogalaxy rows use $d > 1$. ✓: satisfied; ✗: a required condition fails; —: not established here or not applicable.

				Extract-or-Extend (Theorem 2)				Applications			
Protocol	μ	i^*_{ss}	i^*_{zk}	$k_{i\neq i^*_{\mathsf{ss}}}{=}1$	$k_{i^*_{\mathsf{ss}}}{=}h_{i^*_{\mathsf{ss}}}$	Pred.	Applies	WS §4.2	WS §4.3	TDC §5	Fischlin §6
PPR-KKW	2	2	2	✓	✓	none	✓	✓[a]	—[j]	✓	—[l]
C&C-KKW	2	2	2^z	✓	✓	$\boldsymbol{\Phi}$	✗[b]	✗[c]	✓	✗[d]	—[m]
Σ-prot. with helper	2	2	—	✓	✓	$\boldsymbol{\Phi}$	✗[b]	✗[c]	—	✗[d]	—[m]
KZG Boolean check	2	2	—	✓	✓	none	✓	✗[q]	—	✗[e]	✓[r]
Plonk	5	4	3^p	✗[f]	✓	none	✗	✗[h]	—	✗[e]	✗[u]
Sumcheck in Spartan[k]	$2\nu{+}1$	$2\nu{+}1$	—	✓	✓	$\boldsymbol{\Phi}, \Phi_{cr}$	—	—	—	—	—
Protogalaxy	3	3	—	✓	✗[g]	none	✗[g]	✗[g]	—	✗[e]	✓[r]

[a] As a $(1, n)$-WS, since $k_{i^*_{\mathsf{ss}}} = t + 1 = 2$; simulatability of a single transcript is CRSHVZK at round 2. [b] With trivial Φ_{cr}, a critical tree whose two second-round challenges differ in fewer than d coordinates violates $\boldsymbol{\Phi}|_{\vec{k}}$, so condition (1) fails. For $d = 1$, compatibility holds, but $p^{\mathsf{chal}}_{1,2} = \tau/m$ need not be negligible. [c] Theorem 3 requires $\boldsymbol{\Phi}$ to ignore the critical-round challenges, which $\boldsymbol{\Phi}^{(d)}$ reads. [d] $\boldsymbol{\Phi}$ inspects the critical-round challenges, contrary to Theorem 7; the C&C-KKW attack is in Section 1.2. [e] Requires $k_{i^*_{\mathsf{ss}}} = 2$. [f] The critical tree $(1, 1, 1, 4n{+}21, 6)$ branches at rounds 4 and 5. [g] $k_{i^*_{\mathsf{ss}}} = k + 1 < dk + 1 = h_{i^*_{\mathsf{ss}}}$. For $d = 1$, equality holds and Theorem 2 applies, but WS simulatability is not established. [h] The critical round is not the last round. [j] In $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ the new last round is the critical round, so the shared challenge $c_2 \in [n]$ of $\Pi_{\mathsf{PPR.KKW}}$ becomes an ordinary Fiat-Shamir round and contributes a non-negligible $1/n$ to the extraction error. [k] ν is the number of sumcheck iterations, written μ in Section B.4. The sumcheck row refers to the tree-extraction guarantee of Lemma 10. [l] The bound of Theorem 8 contains $2(Q_{P^*} + 1)/n$, non-negligible for polynomial n. [m] These predicate extended-special-soundness parameters are outside the plain setting of Theorem 8. [p] [ABB+26, Proposition 1]. [q] For hard witnesses, $(1, 2n)$-simulatability fails: $n + 1 \leq 2n$ query responses recover both witness polynomials. [r] The RUR proofs are Propositions 1 and 3. Apply Theorem 8 with negligible field-size terms and the stated binding assumptions. [u] This Plonk variant does not satisfy the additional RUR condition of Theorem 8. [z] With hiding commitments, the round-2 simulator of [ABB+26, Theorem 7] applies to each selected execution.

the search cost of the multi-round Fischlin transform of Section 6. Their RUR property in the sense of [RT25, Definition 5] is established in Propositions 1 and 3. The sumcheck sub-protocol of Spartan has the tree-extraction guarantee of Lemma 10. Plonk remains a CRSS example (Proposition 2), but is not covered by the Fischlin application because its RUR requirement is not satisfied. C&C-KKW will further serve as the instantiation of our (t, n)-witness sharing scheme in Section 4.3.

Table 2 records which of these protocols meet the hypotheses of Theorem 2 and of our applications.

4 Witness Sharing

4.1 Definitions

In the senario of witness sharing, a dealer shares a witness w of an NP statement $R(x, w) = 1$ among n recipients such that each share both convinces the recipient of its validity and enables efficient proofs of knowledge of w satisfying $R(x, w) = 1$ at a later stage.

Definition 16. *A (t, n)-WS scheme for relation R consists of three algorithms:*

- $\mathsf{Share}(x, w) \to (com, \sigma_1, \ldots, \sigma_n)$*: On input an instance x and witness w such that $R(x, w) = 1$, it generates a public commitment com and shares $\sigma_1, \ldots, \sigma_n$ for n recipients.*
- $\mathsf{Ver}(x, com, \sigma_i) \to \{0, 1\}$*: On input an instance x, a commitment com, and a share σ_i, it outputs 1 or 0 showing acceptance or rejection, respectively.*
- $\mathsf{Rec}(x, com, \sigma_{i_1}, \ldots, \sigma_{i_{t+1}}) \to w$ *: On input an instance x, a commitment com, and shares $\sigma_{i_1}, \ldots, \sigma_{i_{t+1}}$ from distinct recipients, it outputs w.*

It is correct if for any (x, w) satisfying $R(x, w) = 1$ and any $J \subseteq [n]$ with $|J| = t{+}1$, for $(com, \sigma_1, \ldots, \sigma_n)$ generated by $\mathsf{Share}(x, w)$*, $\mathsf{Ver}(x, com, \sigma_i) = 1$ holds for all $i \in [n]$ and $\mathsf{Rec}(x, com, (\sigma_i)_{i\in J}) = w$ holds with overwhelming probability over the coins of* Share *and the random oracles, if any.*

Security notions for WS schemes include *privacy* and *soundness* similar to those for verifiable secret sharing schemes [Ped92]. For completeness, we present their definitions below.

Definition 17 (Privacy). *A (t,n)-WS scheme for relation R is private if there exists a* PPT *algorithm $\mathcal{S}$ that, for any (x,w) such that $R(x,w)=1$ and any $I \subseteq [n]$ with $|I|=t$, the distributions*

$$D_0 := \{(com, \{\sigma_i\}_{i\in I}) \mid (com, \sigma_1, \ldots, \sigma_n) \leftarrow \mathsf{Share}(x,w)\} \quad \text{and} \quad D_1 := \mathcal{S}(1^\lambda, x, I)$$

are computationally indistinguishable.

Definition 18 (Soundness). *A (t,n)-WS scheme for relation R is sound if for any* PPT *algorithm $\mathcal{A}$, the following probability is negligible in λ:*

$$\Pr\left[\begin{array}{l|l} J \subseteq [n] \wedge |J| = t+1 & \mathcal{A}(1^\lambda) \to (x, com, J, \{\sigma_i\}_{i\in J}), \\ \wedge_{i\in J} \mathsf{Ver}(x, com, \sigma_i) = 1 & \mathsf{Rec}(x, com, \{\sigma_i\}_{i\in J}) \to w \\ \wedge\, R(x,w) \neq 1 & \end{array}\right]$$

When instance x has multiple valid witnesses, different sets of shares may recover different witnesses. Depending on the context, it may be needed to assure that the same witness is recovered from different sets of shares.

Definition 19 (Unique Recovery). *A (t,n)-WS scheme for relation R is unique recovery if, for any* PPT *algorithm $\mathcal{A}$, the following probability is negligible in λ:*

$$\Pr\left[\begin{array}{l|l} \wedge_{i\in\{0,1\}} I_i \subset [n] \wedge |I_i| = t+1 & \mathcal{A}(1^\lambda) \to (x, com, I_0, I_1, \{\sigma_i\}_{i\in[I_0\cup I_1]}), \\ \wedge_{i\in[I_0\cup I_1]} \mathsf{Ver}(x, com, \sigma_i) = 1 & \mathsf{Rec}(x, com, \{\sigma_i\}_{i\in[I_0]}) \to w_0, \\ \wedge_{i\in\{0,1\}}\, R(x, w_i) = 1 & \mathsf{Rec}(x, com, \{\sigma_i\}_{i\in[I_1]}) \to w_1 \\ \wedge w_0 \neq w_1 & \end{array}\right]$$

Unique recovery is a property of the relation rather than of the sharing mechanism. It is added to any sound WS scheme by committing to the witness with a binding and hiding commitment and including the opening check in the relation, and it holds for free when instances have at most one witness; see Corollary 1. Our constructions are therefore presented for soundness only.

4.2 Generic Construction from CRSS Proof System

Let Π be a $2\mu+1$-move $(h_1,\ldots,h_\mu,\boldsymbol{\Phi})$-SS and $(1,\ldots,1,t+1,\Phi_{cr})$-CRSS proof system for R with critical round μ. Since its critical tree branches only in the last round, $t+1$ transcripts with a common prefix recover a witness, enabling one transcript per recipient. We require Π be simulatable in the following sense.

Definition 20 ($(1,\ldots,1,t)$-Simulatability). *A $2\mu+1$-move public-coin proof system Π is called* $(1,\ldots,1,t)$-simulatable *if there exists a* PPT *algorithm $\mathcal{S}$ that, given 1^λ and an instance x as input, outputs a $(1,\ldots,1,t)$-tree of transcripts of Π for x with distribution that is indistinguishable from those generated as follows: Uniformly pick $c_i \leftarrow \mathcal{C}_i$ for $i \in [\mu-1]$, and uniformly pick pairwise distinct $c_\mu^1,\ldots,c_\mu^t \in \mathcal{C}_\mu$. Run the prover of Π with a witness w satisfying $R(x,w)=1$ and challenges $c_1,\ldots,c_{\mu-1}$ up to move $2\mu-1$, and complete the execution t times from the same prover state, once with each challenge c_μ^k, $k\in[t]$, to obtain transcripts $tr_1,\ldots,tr_t$ sharing the prefix $(a,c_1,z_1,\ldots,c_{\mu-1},z_{\mu-1})$. Output the $(1,\ldots,1,t)$-tree of transcripts consisting of $tr_1,\ldots,tr_t$.*

Let C_μ be the challenge space in the μ-th round of Π such that $|C_\mu| \geq n$. Let G be a hash function $\{0,1\}^* \to C_\mu^n$ that maps an input to n distinct challenges in C_μ. We construct a (t,n)-WS scheme using Π and G as in Figure 2.

Theorem 3. *The construction in Figure 2 is a correct, sound, and private (t,n)-WS scheme for relation R if the following hold, with all hash functions modeled as random oracles:*

- *Π is an extended $(h_1,\ldots,h_\mu,\boldsymbol{\Phi})$-SS and $(1,\ldots,1,t+1,\Phi_{cr})$-CRSS argument system for R with critical round μ and hardness assumption on R_{hd}, where $h_\mu = t+1$, Φ_{cr} is trivial (always returns 1), $\boldsymbol{\Phi}$ does not depend on the challenges of round μ, and $\boldsymbol{\Phi}|_{(1,\ldots,1,t+1)}(T^*) = 1$ for every accepting critical tree T^*;*

Parameter: Let Π be a $2\mu+1$-move $(h_1,\ldots,h_\mu,\boldsymbol{\Phi})$-SS and $(1,\ldots,1,t+1,\Phi_{cr})$-CRSS proof system for relation R with critical round μ.

Share: On input (x,w), it runs Π with (x,w) and challenges generated by the Fiat-Shamir heuristic up to the $2\mu-1$ step. Let com be the messages output from the prover so far.
The verifier's challenge in Step 2μ is generated by $G(x,com)\to(c_\mu^1,\ldots,c_\mu^n)$. For each $k\in[n]$, the algorithm continues the execution of Π with challenge c_μ^k in Step $2\mu+1$ and obtains the final output z_μ^k from the prover. The algorithm sets $\sigma_k:=(z_\mu^k,k)$ and outputs $(com,\sigma_1,\ldots,\sigma_n)$.

Ver: On input (x,com,σ_i) for $\sigma_i=(z_\mu,i)$, it interprets (com,z_μ) as a full transcript of Π for instance x and verifies it using the verifier's algorithm with Fiat-Shamir transformation except that the challenge in Step 2μ is set to the i-th element of $G(x,com)$.
It outputs 1 if the verification passes, and 0 otherwise.

Rec: On input $(x,com,\sigma_{i_1},\ldots,\sigma_{i_{t+1}})$, it translates $(com,\sigma_{i_1},\ldots,\sigma_{i_{t+1}})$ to a $(1,\ldots,1,t+1)$-subtree of transcripts of Π for instance x following the Fiat-Shamir transformation and applies the critical-round witness extractor CrExt.
It outputs the extracted witness w.

Fig. 2. Generic construction of (t,n)-WS scheme from CRSS proof system Π.

- *Π is $(1,\ldots,1,t)$-simulatable;*
- *on an accepting $(1,\ldots,1,t+1)$-tree generated by the honest prover with witness w, CrExt outputs w whenever it outputs a valid witness;*
- *$1/|\mathcal{C}_i|$ is negligible for every $i\in[\mu-1]$, $|\mathcal{C}_\mu|\geq n$, and every failure density of $\boldsymbol{\Phi}$ is negligible.*

Proof. Let H be the random oracle that provides the Fiat-Shamir challenges of rounds $1,\ldots,\mu-1$, evaluated on x and the prefix of the transcript as in FS$[\Pi]$, and let G be the random oracle of Step 2μ. For $\sigma_i=(z_\mu^i,i)$, the pair (com,z_μ^i) is a transcript of Π whose round-μ challenge is the i-th entry of $G(x,com)$. Since the entries of $G(x,com)$ are pairwise distinct, any $t+1$ such transcripts with distinct indices i form a $(1,\ldots,1,t+1)$-tree of Π, and Rec runs CrExt on it.

Correctness. Let $(com,\sigma_1,\ldots,\sigma_n)\leftarrow\mathsf{Share}(x,w)$. Each (com,z_μ^k) is an accepting transcript by the completeness of Π, so $\mathsf{Ver}(x,com,\sigma_k)=1$ for all $k\in[n]$. Let $J\subseteq[n]$ with $|J|=t+1$ and let T^* be the tree that Rec assembles from $(\sigma_i)_{i\in J}$. By the third condition of the theorem, $\mathsf{Rec}(x,com,(\sigma_i)_{i\in J})\neq w$ only if $\mathsf{CrExt}(x,T^*)$ outputs no valid witness. Regard Share followed by the selection of J as an adversary that outputs T^*: its challenges of rounds $i<\mu$ are answers of H, and its round-μ challenges are chosen without querying H. Since $h_\mu=t+1$, Φ_{cr} is trivial, and $\boldsymbol{\Phi}|_{(1,\ldots,1,t+1)}(T^*)=1$, the main claim of Theorem 2 applies with $i^*=\mu$ and bounds the probability that $\mathsf{CrExt}(x,T^*)$ fails by $\kappa_{ex}+\epsilon_{hd}\theta$, with round μ removed from κ_{ex} and θ. This is negligible: κ_{ex} is a sum of terms $(h_i-1)/|\mathcal{C}_i|$ and failure densities of $\boldsymbol{\Phi}$ for $i\in[\mu-1]$, each negligible, $\theta\leq1$, and ϵ_{hd} is negligible by the hardness of R_{hd}.

Soundness. Let $\mathcal{A}$ output $(x,com,J,\{\sigma_i\}_{i\in J})$ with $J\subseteq[n]$, $|J|=t+1$, and $\mathsf{Ver}(x,com,\sigma_i)=1$ for all $i\in J$. Then Rec obtains an accepting $(1,\ldots,1,t+1)$-tree T^* of Π and outputs $w\leftarrow\mathsf{CrExt}(x,T^*)$. Let $\mathcal{A}'$ run $\mathcal{A}$ and output T^* (or $\perp$ if some share fails to verify). Its challenges of rounds $i<\mu$ are answers of H, and its round-μ challenges are pairwise distinct values chosen by itself, as $t+1$ entries of one answer of G. Hence $\mathcal{A}'$ is an adversary in the main claim of Theorem 2 with $i^*=\mu$. The condition $h_\mu=k_\mu=t+1$ and compatibility follow from the hypotheses of Theorem 3, so Theorem 2 gives

$$\Pr\Big[\textstyle\bigwedge_{i\in J}\mathsf{Ver}(x,com,\sigma_i)=1\ \wedge\ R(x,w)\neq1\Big]\leq\Pr\big[R(x,w)=0\wedge\Phi_{cr}(T^*)=1\big]<\kappa_{ex}+\epsilon_{hd}\theta,$$

where Q counts the H-queries of $\mathcal{A}$ and round μ is removed from κ_{ex} and θ. The right-hand side is negligible as in the correctness argument. The instance x is chosen by $\mathcal{A}$; the tree builder behind Theorem 2 handles adaptively chosen instances as in Theorem 1.

Privacy. On input $(1^\lambda,x,I)$ with $|I|=t$, the simulator runs the $(1,\ldots,1,t)$-simulator of Π on x to obtain a tree with prefix $(a,c_1,z_1,\ldots,c_{\mu-1},z_{\mu-1})$ and t leaves (c_μ^k,z_μ^k), $k\in[t]$. Let $\pi:I\to[t]$ be the order-preserving bijection. The simulator sets $com:=(a,z_1,\ldots,z_{\mu-1})$, programs H so that the round-i challenge of the prefix $(x,a,z_1,\ldots,z_{i-1})$ is c_i for $i\in[\mu-1]$, and programs $G(x,com)$ to the

n-tuple whose i-th entry is $c_\mu^{\pi(i)}$ for $i \in I$ and whose remaining $n-t$ entries are uniform pairwise distinct elements of $\mathcal{C}_\mu \setminus \{c_\mu^1, \ldots, c_\mu^t\}$. It outputs com and $\sigma_i := (z_\mu^{\pi(i)}, i)$ for $i \in I$. In D_0, the honest prover queries H at the $\mu - 1$ prefixes above and G at (x, com), all distinct points, and the answers are uniform: $c_1, \ldots, c_{\mu-1}$ uniform in their challenge spaces and $G(x, com)$ uniform among n-tuples of distinct elements. Hence the joint distribution of $(com, \{\sigma_i\}_{i \in I})$ and the two oracles in D_0 is obtained from the real distribution of Definition 20, with its t round-μ challenges placed at the positions in I and padded to a uniform distinct n-tuple, by setting the oracles to these values at the prover's query points and to uniform values elsewhere. D_1 is obtained by the same procedure from the simulated tree. A distinguisher between D_0 and D_1 with oracle access therefore distinguishes the real distribution of Definition 20 from the simulated one, contradicting the $(1, \ldots, 1, t)$-simulatability of Π. □

Corollary 1 (Unique recovery). *Let* Com *be a computationally binding and hiding commitment scheme, and let* R' *be the relation with* $R'((x,c),(w,r)) = 1$ *if and only if* $R(x,w) = 1$ *and* $\mathsf{Com}(w;r) = c$. *Let* WS' *be a correct, sound, and private* (t,n)*-WS scheme for* R'. *The* (t,n)*-WS scheme for* R *in which* $\mathsf{Share}(x,w)$ *picks* r, *runs* $\mathsf{WS}'.\mathsf{Share}((x,c),(w,r))$ *for* $c := \mathsf{Com}(w;r)$, *and publishes* c *as part of the commitment, while* Ver *and* Rec *run those of* WS' *on* (x,c) *and* Rec *outputs the* w*-component of the recovered witness, is correct, sound, private, and unique recovery.*

Proof. Correctness and soundness carry over from WS' since $R'((x,c),(w,r)) = 1$ implies $R(x,w) = 1$. For privacy, the simulator picks r, sets $c := \mathsf{Com}(0;r)$, and runs the simulator of WS' on $((x,c), I)$. The real distribution is first replaced by $c := \mathsf{Com}(w;r)$ together with the output of the simulator of WS' on $((x,c), I)$, which is indistinguishable by the privacy of WS' for the instance (x,c) with witness (w,r), and then c is replaced by $\mathsf{Com}(0;r)$, which is indistinguishable by hiding since the simulator of WS' does not use w or r. For unique recovery, let two quorums recover (w_0, r_0) and (w_1, r_1). By the soundness of WS', both satisfy R' with overwhelming probability, so $\mathsf{Com}(w_0;r_0) = \mathsf{Com}(w_1;r_1) = c$, and $w_0 \neq w_1$ breaks the binding of Com. □

In the random oracle model, $\mathsf{Com}(w;r) := H(w,r)$ with a random salt r is binding and hiding, so the hash-only instantiation of Section 4.3 obtains unique recovery at the cost of one hash evaluation inside the relation.

Direct instantiations. For $t = 1$ the conditions of Theorem 3 are met by every 2-special-sound honest-verifier zero-knowledge Σ-protocol with a challenge space of size at least n, e.g., Schnorr's protocol: its critical tree consists of two transcripts, its predicates are trivial, and (1)-simulatability is honest-verifier zero-knowledge since a (1)-tree is a single transcript. Such protocols therefore yield $(1,n)$-WS schemes directly. Larger thresholds are the subject of the next subsection.

4.3 Construction from MPCitH

Two conditions of Theorem 3 are restrictive. First, $(1, \ldots, 1, t)$-simulatability asks for t simulated transcripts sharing a common prefix, whereas zero-knowledge simulators generally produce only a single transcript, which limits the direct approach to $t = 1$. Second, since the round-μ challenges of the shares are taken from G rather than from the Fiat-Shamir oracle, Theorem 3 requires the extended tree to have exactly $t+1$ branches at round μ and the predicate system $\boldsymbol{\Phi}$ to ignore the round-μ challenges. Extended predicate systems arising in practice, such as $\boldsymbol{\Phi}^{(d)}$ of C&C-KKW in Section B.1, read the challenges of the critical round for every d with negligible failure density, so C&C-KKW cannot be plugged in directly even for $t = 1$. We therefore add a threshold layer to an MPCitH protocol. The resulting proof system has a new last round that is the critical round with exactly $t+1$ branches, inherits the predicate system on the truncated tree so that the new round is ignored, and is $(1, \ldots, 1, t)$-simulatable for any t.

Abstract MPC structure. Consider an MPC protocol for computing $R(x,w) = 1$ with m players each holding a share w_i. We assume that it consists of two functionalities and a verification algorithm:

- $\mathsf{Prep}((\xi_i)_{i\in[m]}) \to (r_i)_{i\in[m]}$: a witness-independent preprocessing phase where each party $i \in [m]$ is given random seed ξ_i as input and outputs correlated randomness r_i.
- $\mathsf{ExecMPC}((s_i)_{i\in[m]}) \to (msg_i)_{i\in[m]}$: an online phase where, party $i \in [m]$ is given $s_i = (w_i, r_i)$ as input, and outputs msg_i that consists of messages to other parties and the final output of the computation.

– $\mathsf{VerMPC}(\hat{i}, msg_{\hat{i}}, (s_i)_{i\in[m]\setminus\{\hat{i}\}}) \to \{0,1\}$: a polynomial-time verification algorithm that is given the index $\hat{i} \in [m]$ of one party, its output $msg_{\hat{i}}$, and the private inputs s_i of all other parties, and outputs 1 if and only if $msg_{\hat{i}}$ contains 1 as the final output and every party $i \in [m] \setminus \{\hat{i}\}$ holding s_i reaches final output 1 when party $\hat{i}$ outputs $msg_{\hat{i}}$.

Abstract MPCitH Scheme. The MPCitH framework constructs a $(2\mu+1)$-move public-coin proof system Π_{MPCitH} using the MPC protocol as follows:

– Steps 1 to $2\mu-2$: The prover executes Prep several times, and the verifier agrees on one of the executions that proceed to the online phase.[8] Let $\mathsf{Prep}(\{\xi_i\}_{i\in[m]}) \to (\{r_i\}_{i\in[m]})$ denote the selected execution. The prover commits to every r_i.
– Step $2\mu-1$: The prover additively share witness w into $\{w_i\}_{i\in[m]}$ and runs $\mathsf{ExecMPC}(\{s_i\}_{i\in[m]}) \to \{msg_i\}_{i\in[m]}$ with $s_i := (w_i, r_i)$. The prover commits to every s_i and msg_i.
– Step 2μ: The verifier sends challenge $\hat{i} \leftarrow [m]$.
– Step $2\mu+1$: The prover opens $\{s_i\}_{i\in[m]\setminus\{\hat{i}\}}$ and $msg_{\hat{i}}$.
– Final Step: The verifier checks opened values with respective commitments, and verifies their consistency by running $\mathsf{VerMPC}(\hat{i}, msg_{\hat{i}}, (s_i)_{i\in[m]\setminus\{\hat{i}\}})$. It accepts if all checks pass.

Merkle trees are used for commitments in the above protocol to reduce communication overhead. We require the Merkle-tree scheme to be hiding, where each leaf contains a commitment to an actual message. In the random oracle model, this can be instantiated by hashing each message on a leaf together with a fresh random salt of sufficient length. When a leaf is opened, the corresponding salt (or the commitment opening information in general) is revealed as well. For notational simplicity, we omit the salts from the notation.

To amplify soundness, the above protocol is typically repeated multiple times in parallel, merging the preprocessing part to gain efficiency. We describe a single execution for readability and explain, after the threshold construction below, how the repetitions are handled and what they cost.

For Π_{MPCitH}, we consider a CRSS extractor of the following form. Let K^* be a $(1,\dots,1,k_\mu)$-tree of transcripts with $k_\mu \geq 2$ satisfying Φ_{cr}. Its transcripts conceal pairwise distinct virtual players, so every private input s_i is opened in at least one of them. We call K^* *consistent* if every virtual player that is opened in more than one transcript is opened to the same value; a consistent K^* thus determines $\{s_i\}_{i\in[m]}$ with $s_i = (w_i, r_i)$. A *special* CRSS extractor $\mathsf{CrExt}^*_{\mathsf{MPCitH}}$ is one that, given a consistent K^*, outputs a witness w *only if* the witness w is determined by the opened private inputs w_i. Although we call it a special extractor, this condition is very mild: the canonical MPCitH extractor reconstructs w exactly in this way, as does the critical extractor of C&C-KKW in Section B.1 for each MPC execution whose views are fully opened.

Adding threshold structure. We now construct $2\mu+3$-move proof system $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ whose critical round is $\mu+1$. Let q be a prime that $|q| > |s_i|$ for all $i \in [m]$.

– The prover and verifier execute Π_{MPCitH} to the end. In addition, they do the following procedure in parallel from Step $2\mu-1$ of Π_{MPCitH}.
– Step $2\mu-1$: For each $i \in [m]$, the prover picks random seed ρ_i and use it to generate a degree-t pseudo-random polynomial $f_i(X) \in \mathbb{F}_q[X]$ satisfying $f_i(0) = s_i$, and compute shares $s_{i,k} := f_i(k)$ for $k \in [n]$. The prover computes two Merkle tree roots rt_x over the shares $s_{i,k}$ for $k \in [n]$, $i \in [m]$, and rt_ρ over the random seeds ρ_i for $i \in [m]$. The prover sends rt_x and rt_ρ to the verifier.
– Step 2μ: The verifier's challenge $\hat{i} \in [m]$ is taken from the same step of Π_{MPCitH}.
– Step $2\mu+1$: The prover opens ρ_i for all $i \in [m] \setminus \{\hat{i}\}$ with respect to MT root rt_ρ. It also opens co-path of $s_{i,k}$ for all $i \in [m] \setminus \{\hat{i}\}$ and $k \in [n]$ with respect to MT root rt_x.
– Step $2\mu+2$: The verifier sends a random $k \in [n]$ to the prover.
– Step $2\mu+3$: The prover opens share $s_{\hat{i},k}$ with respect to MT root rt_x.
– Final Step: For each opened virtual player $i \in [m] \setminus \{\hat{i}\}$, the verifier checks the consistency of opened ρ_i's with respect to rt_ρ and recovers the sharing polynomial $f_i(X)$ from ρ_i and s_i opened in Step $2\mu+1$ of Π_{MPCitH}. The verifier then computes all shares $s_{i,1},\dots,s_{i,n}$ and verifies their consistency with respect to rt_x using the co-paths given in Step $2\mu+1$. For the concealed virtual player $\hat{i}$, the verifier verifies the consistency of $s_{\hat{i},k}$ opened in Step $2\mu+3$ with respect to rt_x. It accepts if all checks pass.

[8] To be precise, the preprocessing phase ends at the $(2\mu-1)$-th move, which is merged with the first step of the online phase.

Here, ρ_i can be generated via a GGM tree, hence to open all ρ_i except for $i = \hat{i}$, it requires $O(\log m)$ bits of communication.

Parallel repetition. A single MPC execution has round-μ challenge space $[m]$ and hence soundness error at least $1/m$. As usual, Π_{MPCitH} is run τ times in parallel with independent challenges $(\hat{i}^{(1)}, \ldots, \hat{i}^{(\tau)}) \in [m]^\tau$ in Step 2μ, so that the error decays as $m^{-\tau}$. C&C-KKW in Section B.1 achieves a comparable amplification, up to the cut-and-choose soundness bound, while sharing the preprocessing phase among the executions. The threshold layer is applied to every execution $e \in [\tau]$ separately, with its own polynomials $f_i^{(e)}$ and seeds $\rho_i^{(e)}$, while rt_x and rt_ρ are single Merkle roots over all executions. The challenge $k \in [n]$ of Step $2\mu+2$, in contrast, is *not* repeated: it is common to all executions, and the prover opens the k-th share of the concealed player of every execution, as the k-th recipient of Figure 2 must reconstruct every concealed input together with any t others. The protocol thus has the same partial-repetition structure as $\Pi_{\mathsf{PPR\text{-}KKW}}$ in Section 3.3: the challenge spaces up to round μ are exponentially large, whereas the critical round $\mu+1$ keeps challenge space $[n]$ regardless of τ. This suffices, since Theorem 3 requires $1/|\mathcal{C}_i|$ to be negligible only before the critical round and merely $|\mathcal{C}_{\mu+1}| \geq n$ at the critical round. Theorems 4 and 5 hold verbatim, as their proofs treat the leaves of rt_x and the polynomials f_i individually; in particular, t transcripts reveal only t evaluations of each degree-t polynomial, so privacy is unaffected by τ.

Security. Let $\mathsf{Truncate}$ be a function that takes a transcript of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ and removes the last two moves and relevant commitments, resulting in a transcript of Π_{MPCitH}. We abuse notation and also use $\mathsf{Truncate}$ for trees of transcripts, i.e., it extracts a tree of transcripts of Π_{MPCitH} from a tree of transcripts of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$. Let $\boldsymbol{\Phi}$ be a predicate system over $(h_1, \ldots, h_\mu)$-trees of transcripts of Π_{MPCitH}. Let Φ_{cr} be a predicate over $(1, \ldots, 1, k_\mu)$-subtrees of transcripts of Π_{MPCitH}. By $\boldsymbol{\Phi}^+$, we denote a predicate system over $(h_1, \ldots, h_\mu, t+1)$-trees of transcripts of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ defined as follows. A tree of transcripts T of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ satisfies $\boldsymbol{\Phi}^+$ if and only if $\mathsf{Truncate}(T)$ satisfies $\boldsymbol{\Phi}$ and its all $(1, \ldots, 1, k_\mu)$-subtrees satisfy Φ_{cr}. Let ϕ_ϵ be an empty predicate that always returns true. The following theorem states the security of the above modified proof system.

Theorem 4. *If Π_{MPCitH} is an extended $(h_1, \ldots, h_\mu, \boldsymbol{\Phi})$-SS and $(1, \ldots, 1, k_\mu, \Phi_{cr})$-CRSS argument system with a special extractor $\mathsf{CrExt}^*_{\mathsf{MPCitH}}$ for a relation R and hardness assumption R_{hd}, then $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ is an extended $(h_1, \ldots, h_\mu, t+1, \boldsymbol{\Phi}^+)$-SS and $(1, \ldots, 1, t+1, \phi_\epsilon)$-CRSS proof system with critical round $\mu+1$ with hardness assumption on R_{hd} and the binding property of the Merkle-tree scheme where $\boldsymbol{\Phi}^+$ is a predicate system that is derived from $\boldsymbol{\Phi}$ and Φ_{cr} as described above.*

Proof. Throughout, $n \geq t+1$ and $k_\mu \geq 2$: the former is needed for a $(1, \ldots, 1, t+1)$-tree of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ to exist, since its last-round challenges are pairwise distinct elements of $[n]$, and the latter is part of the CRSS hypothesis on Π_{MPCitH} at critical round μ. We recall two facts about the verifier of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ that the proof relies on. For every opened virtual player $i \neq \hat{i}$, the verifier recomputes the polynomial f_i of degree at most t from the opened ρ_i and s_i, with $f_i(0) = s_i$, and verifies *all* n leaves (i, k), $k \in [n]$, of rt_x against $f_i(k)$. For the concealed player $\hat{i}$, only the single leaf $(\hat{i}, k)$ selected in Step $2\mu+2$ is verified against rt_x, and no degree check is performed.

We construct a critical-round extractor and an extended extractor as follows.

Critical-round extractor. Let T_0^* be an accepting $(1, \ldots, 1, t+1)$-tree of transcripts of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$. Its $t+1$ transcripts share the prefix up to Step $2\mu+1$, hence the roots rt_x, rt_ρ, the concealed virtual player $\hat{i}_0$, and the openings of s_i and ρ_i for $i \neq \hat{i}_0$. They differ only in the pairwise distinct challenges $k_1, \ldots, k_{t+1} \in [n]$ of Step $2\mu+2$ and in the shares $s_{\hat{i}_0, k_\ell}$ opened in Step $2\mu+3$. In particular, $\mathsf{Truncate}(T_0^*)$ is a single accepting transcript of Π_{MPCitH}, which we denote by tr_0^*. $\mathsf{CrExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T_0^*)$ proceeds as follows.

1. Collect s_i for all $i \in [m] \setminus \{\hat{i}_0\}$ opened in tr_0^*.
2. Let $g \in \mathbb{F}_q[X]$ be the unique polynomial of degree at most t with $g(k_\ell) = s_{\hat{i}_0, k_\ell}$ for all $\ell \in [t+1]$, and set $s_{\hat{i}_0} := g(0)$.
3. Parse $s_i = (w_i, r_i)$ for all $i \in [m]$ and reconstruct the witness w from the private inputs of all virtual players exactly as $\mathsf{CrExt}^*_{\mathsf{MPCitH}}$ does. Output w if $R(x, w) = 1$, and $\perp$ otherwise.

We call $\{s_i\}_{i \in [m]}$ computed in the first two steps the private inputs *recovered* from T_0^*.

Extended extractor. Let T be an accepting $(h_1, \ldots, h_\mu, t+1)$-tree of transcripts of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ with $\boldsymbol{\Phi}^+(T) = 1$, and let T_0^* be its leftmost $(1, \ldots, 1, t+1)$-subtree, so that $T \sqsupseteq T_0^*$. $\mathsf{ExExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T)$ first runs $\mathsf{CrExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T_0^*)$ and outputs w if $w \neq \perp$. Otherwise it proceeds as follows.

Let T' be the $(1, \ldots, 1, k_\mu, t+1)$-subtree of T formed by the first k_μ branches at round μ. It contains T_0^* and $k_\mu - 1$ further $(1, \ldots, 1, t+1)$-subtrees $T_1^*, \ldots, T_{k_\mu - 1}^*$. All transcripts in T' share the prefix up to Step $2\mu - 1$, hence rt_x, rt_ρ, and the commitments of Π_{MPCitH}, and they differ in the challenge of Step 2μ: T_j^* conceals virtual player $\hat{i}_j$, where $\hat{i}_0, \ldots, \hat{i}_{k_\mu - 1}$ are pairwise distinct. For every $j \in [k_\mu - 1]$, $\mathsf{ExExt}^{\mathsf{th}}_{\mathsf{MPCitH}}$ recovers private inputs $\{s_i^{(j)}\}_{i \in [m]}$ from T_j^* in the same way as $\mathsf{CrExt}^{\mathsf{th}}_{\mathsf{MPCitH}}$ does, and we write $\{s_i^{(0)}\}_{i \in [m]}$ for those recovered from T_0^*. Let $f_i^{(j)}$ denote the polynomial recomputed by the verifier for opened player $i \neq \hat{i}_j$ in T_j^*, so that $f_i^{(j)}(0) = s_i^{(j)}$.

Case 1: $s_i^{(j)} \neq s_i^{(0)}$ for some $j \in [k_\mu - 1]$ and $i \in [m]$. We show that T_0^* and T_j^* contain two verified openings of one leaf of rt_x to distinct values, which $\mathsf{ExExt}^{\mathsf{th}}_{\mathsf{MPCitH}}$ outputs as a break of the binding property of the Merkle-tree scheme. There are three sub-cases.

- $i \notin \{\hat{i}_0, \hat{i}_j\}$. Player i is opened in both subtrees, so all n leaves (i, k) are verified against $f_i^{(0)}(k)$ in T_0^* and against $f_i^{(j)}(k)$ in T_j^*. Since $f_i^{(0)}(0) = s_i^{(0)} \neq s_i^{(j)} = f_i^{(j)}(0)$, the two polynomials are distinct and of degree at most t, hence they agree on at most t points. As $n \geq t+1$, there is $k \in [n]$ with $f_i^{(0)}(k) \neq f_i^{(j)}(k)$.
- $i = \hat{i}_0$. Since $\hat{i}_j \neq \hat{i}_0$, player $\hat{i}_0$ is opened in T_j^*, so every leaf $(\hat{i}_0, k)$, $k \in [n]$, is verified against $f_{\hat{i}_0}^{(j)}(k)$ in T_j^*, where $f_{\hat{i}_0}^{(j)}(0) = s_{\hat{i}_0}^{(j)}$. In T_0^*, the shares $s_{\hat{i}_0, k_\ell}$, $\ell \in [t+1]$, are verified as leaves $(\hat{i}_0, k_\ell)$. If $s_{\hat{i}_0, k_\ell} = f_{\hat{i}_0}^{(j)}(k_\ell)$ held for all $\ell \in [t+1]$, then the interpolated polynomial g of degree at most t would coincide with $f_{\hat{i}_0}^{(j)}$ on $t+1$ points, hence $g = f_{\hat{i}_0}^{(j)}$ and $s_{\hat{i}_0}^{(0)} = g(0) = s_{\hat{i}_0}^{(j)}$, contradicting the case assumption. Hence some leaf $(\hat{i}_0, k_\ell)$ is verified to two distinct values.
- $i = \hat{i}_j$. Symmetric to the previous sub-case with the roles of T_0^* and T_j^* exchanged.

Note that the verifier never checks that the shares of the concealed player lie on a polynomial of degree t; the argument only uses that the other subtree T_j^* opens player $\hat{i}_0$ and thereby fixes all its leaves under rt_x.

Case 2: $s_i^{(j)} = s_i^{(0)}$ for all $j \in [k_\mu - 1]$ and $i \in [m]$. Write $s_i := s_i^{(0)}$ and $s_i = (w_i, r_i)$. For $j \in \{0, \ldots, k_\mu - 1\}$, let $tr_j := \mathsf{Truncate}(T_j^*)$, i.e., the transcript of Π_{MPCitH} obtained by removing the last two moves and the relevant commitments from T_j^*. These are accepting transcripts of Π_{MPCitH} with a common prefix up to Step $2\mu - 1$ and pairwise distinct challenges $\hat{i}_j$ in Step 2μ, so they form an accepting $(1, \ldots, 1, k_\mu)$-tree K^*, which consists of the first k_μ branches at round μ of $K := \mathsf{Truncate}(T)$; that is, $K \sqsupseteq K^*$. In tr_j, the private inputs opened are $\{s_i^{(j)}\}_{i \neq \hat{i}_j} = \{s_i\}_{i \neq \hat{i}_j}$. Hence every virtual player that is opened in several transcripts of K^* is opened to the same value, and the private inputs it determines are exactly $\{s_i\}_{i \in [m]}$. Since $\mathsf{CrExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T_0^*)$ output $\perp$ as a result of executing the same recovery procedure as the special extractor, $\mathsf{CrExt}^{*}_{\mathsf{MPCitH}}(K^*)$, does, $\mathsf{CrExt}^{*}_{\mathsf{MPCitH}}(K^*)$ does not output a valid witness from the collected w_i's. Since $\boldsymbol{\Phi}^+(T) = 1$, K satisfies $\boldsymbol{\Phi}$ and K^* satisfies Φ_{cr}. Applying Definition 14 to Π_{MPCitH} with the critical tree K^* and the extended tree $K \sqsupseteq K^*$, $\mathsf{ExExt}_{\mathsf{MPCitH}}(K)$ outputs s with $R_{hd}(y, s) = 1$. $\mathsf{ExExt}^{\mathsf{th}}_{\mathsf{MPCitH}}$ runs $\mathsf{ExExt}_{\mathsf{MPCitH}}(K)$ and outputs whatever it outputs.

For extended special soundness, let T be any accepting $(h_1, \ldots, h_\mu, t+1)$-tree with $\boldsymbol{\Phi}^+(T) = 1$. Then $\mathsf{ExExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T)$ outputs a valid witness, two distinct openings of a leaf of rt_x (Case 1), or the output of $\mathsf{ExExt}_{\mathsf{MPCitH}}(K)$, which by the extended $(h_1, \ldots, h_\mu, \boldsymbol{\Phi})$-SS of Π_{MPCitH} is a valid witness or a solution to R_{hd}. Hence $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ is extended $(h_1, \ldots, h_\mu, t+1, \boldsymbol{\Phi}^+)$-SS with respect to the hardness of R_{hd} and the binding of the Merkle-tree scheme. For critical-round special soundness at round $\mu + 1$, the tree shape $(1, \ldots, 1, t+1)$ satisfies $k_i = 1$ for $i \leq \mu$ and $k_{\mu+1} = t+1 = h_{\mu+1}$, and the predicate ϕ_ϵ is trivial. For any accepting $(1, \ldots, 1, t+1)$-tree T_0^*, either $\mathsf{CrExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T_0^*)$ outputs a valid witness, or, for every $T \sqsupseteq T_0^*$ with $\boldsymbol{\Phi}^+(T) = 1$, $\mathsf{ExExt}^{\mathsf{th}}_{\mathsf{MPCitH}}(T)$ outputs a break of the Merkle-tree binding (Case 1) or a solution to R_{hd} (Case 2). This is exactly the requirement of Definition 14. □

Theorem 5. *If Π_{MPCitH} is zero-knowledge, the Merkle-tree scheme is hiding, and the coefficients of the sharing polynomials f_i are derived from the seeds ρ_i by a secure pseudorandom generator* PRG*, then $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ is $(1, \ldots, 1, t)$-simulatable.*

Proof. Write $f_i(X) = s_i + \sum_{j=1}^{t} a_{i,j} X^j$ with $(a_{i,1}, \ldots, a_{i,t}) := \mathsf{PRG}(\rho_i)$. A $(1, \ldots, 1, t)$-tree of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ consists of t transcripts sharing the prefix up to Step $2\mu+1$, hence the concealed player $\hat{i}$, and differing only in the last-round challenges $k_1, \ldots, k_t \in [n]$ and the opened leaves $(\hat{i}, k_\ell)$.

We first construct the simulator $\mathcal{S}$ as follows. On input x, it runs the zero-knowledge simulator of Π_{MPCitH} to obtain $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$, where $c_\mu = \hat{i}$ and z_μ contains $\{s_i\}_{i \neq \hat{i}}$ and $msg_{\hat{i}}$. It picks $s_{\hat{i}} \leftarrow \mathbb{F}_q$ and follows the prover of $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$ honestly from Step $2\mu - 1$ on with $\{s_i\}_{i \in [m]}$: it picks ρ_i, computes f_i, the shares $s_{i,k}$, the roots rt_x, rt_ρ, and the openings of Step $2\mu+1$ for $i \neq \hat{i}$. It then picks pairwise distinct $k_1, \ldots, k_t \leftarrow [n]$ and, for each $\ell \in [t]$, appends k_ℓ and the opening of leaf $(\hat{i}, k_\ell)$ to obtain tr_ℓ. The simulated $s_{\hat{i}}$ need not match the commitment to $s_{\hat{i}}$ in the simulated base transcript, since neither is ever opened.

Now we argue that the output from $\mathcal{S}$ is indistinguishable from the real execution as defined in Definition 20. Let H_0 be the real distribution of Definition 20. Exactly $n-t+1$ leaves are never opened: the leaf of $\rho_{\hat{i}}$ under rt_ρ and the leaves $(\hat{i}, k)$, $k \notin \{k_1, \ldots, k_t\}$, under rt_x. Their leaf commitments appear in the transcripts only inside co-paths.

- H_1: as H_0, but the $n - t + 1$ unopened leaves commit to fixed dummy values. $H_0 \approx H_1$ by the hiding of the leaf commitments of the Merkle-tree scheme, via a hybrid over these leaves.
- H_2: as H_1, but $(a_{\hat{i},1}, \ldots, a_{\hat{i},t}) \leftarrow \mathbb{F}_q^t$. As $\rho_{\hat{i}}$ is used nowhere else in H_1, $H_1 \approx H_2$ by the security of PRG.
- H_3: as H_2, but $s_{\hat{i}} \leftarrow \mathbb{F}_q$ replaces the real private input in $f_{\hat{i}}$. The value $s_{\hat{i}}$ now enters the threshold layer only through the opened shares $f_{\hat{i}}(k_\ell) = s_{\hat{i}} + \sum_j a_{\hat{i},j} k_\ell^j$, $\ell \in [t]$. For pairwise distinct nonzero $k_1, \ldots, k_t$, the matrix $(k_\ell^j)_{\ell,j \in [t]}$ is invertible, so these t shares are uniform in $\mathbb{F}_q^t$ and independent of $s_{\hat{i}}$. Hence $H_2 = H_3$.
- H_4: as H_3, but the base transcript $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ is generated by the simulator of Π_{MPCitH}. The rest of H_3 is computed from $\hat{i}$ and $\{s_i\}_{i \neq \hat{i}}$ read from z_μ and a random $s_{\hat{i}}$, so a distinguisher between H_3 and H_4 breaks the zero-knowledge of Π_{MPCitH}.
- H_5, H_6: undo H_2 and H_1, i.e., derive $(a_{\hat{i},j})_j$ from $\rho_{\hat{i}}$ by PRG again and commit to the actual values in the unopened leaves. $H_4 \approx H_5 \approx H_6$ by the same two reductions.

H_6 is exactly the output distribution of $\mathcal{S}$, which completes the proof. □

From Theorems 4, 5, and 3, we obtain a (t, n)-WS scheme from $\Pi^{\mathsf{th}}_{\mathsf{MPCitH}}$. Since the KKW MPCitH framework [KKW18] satisfies the required properties of zero-knowledge and CRSS (as shown in Section 3), and both KKW and Merkle trees can be implemented only with hash functions modeled as random oracles, we obtain a hash-based (t, n)-WS scheme that is secure under the random oracle model alone.

Performance. Several optimizations should be possible, e.g., sharing private inputs chunk-wise over a smaller field or batching authentication paths across executions, but we restrict ourselves to giving intuition here. The metric that matters for a dealer serving n recipients is the communication per recipient, $|com| + |\sigma_k|$, where com is common to all recipients and σ_k is individual. In the generic construction of Section 4.2, com is a Fiat-Shamir proof of Π truncated before its last round and σ_k is a single last-round response, so a recipient receives no more than one proof, and the dealer runs the prover once plus n last-round completions. In the hash-based instantiation above, com is a τ-fold MPCitH proof plus about $\tau\lambda(\log_2 m + 1)$ bits for the threshold layer, and σ_k consists of τ elements of $\mathbb{F}_q$ with their authentication paths, about $\tau(\log_2 q + \lambda\lceil \log_2 n \rceil)$ bits, independently of t.

4.4 Comparison with Alternative Approaches

In this subsection, we compare our WS scheme with several alternative approaches for distributing a witness w satisfying $R(x, w) = 1$ while enabling verification and threshold-based reconstruction. We discuss their performance characteristics, security properties, and practical challenges, particularly for post-quantum security.

- **VSS + zkSNARG:** The dealer shares w among n parties using a non-interactive verifiable secret sharing scheme VSS, and publishes a non-interactive zero-knowledge proof $\pi := \mathsf{zkSNARG}\{(w) | R(x, w) = 1 \wedge com = \mathsf{VSS.Com}(w)\}$ where com is a commitment published through VSS.

– **DKG + tPKE + zkSNARG:** The n parties first run threshold DKG and publish the resulting encryption key pk to the prover. Prover encrypts w under key pk and generates a proof $\pi := \mathsf{zkSNARG}\{(w)|R(x,w) = 1 \wedge ct = \mathsf{tPKE.Enc}_{pk}(w)\}$. Each party receives c and π.
– **MPCitH + Byzantine Agreement (BA):** The dealer runs (t,n)-threshold MPC for $t < n/3$ in the head among n virtual players each holding a share of witness w. The view of virtual player i is privately sent to each real party i. The parties collaboratively check the consistency of the received view and agree on acceptance or rejection. If they accept, w can be reconstructed from any $t+1$ consistent views.

The first approach is efficient when VSS and zkSNARG admit compatible instantiations, as in the classical constructions of [Ped92, Gro16, CGM16]. Post-quantum lattice-based candidates exist, e.g., [EBM12, DG16, PNSN26, GMNO18, Nit19, BS23, GNS23], but securely composing them under a common assumption can entail large common reference strings, substantial preprocessing, and delicate parameter choices. Although hash-based zkSNARGs are known [AHIV17], VSS typically requires homomorphic commitments that are difficult to realize in the minicrypt setting. By contrast, our WS combines SS with MPCitH and embeds share verification in its commit-and-open structure, enabling an instantiation based solely on hash functions modeled as random oracles.

The second approach encrypts the witness w and broadcasts the ciphertext together with a proof of encryption consistency, thereby achieving efficient communication and public verifiability where *com* itself constitutes a non-interactive proof of knowledge of w. For large witnesses, such as images or videos, however, proving consistency may be costly because the encryption circuit is correspondingly large; for a statement of circuit size C, zkSNARG proving typically requires $O(C \log C)$ computation. Moreover, post-quantum instantiation remains challenging both for the individual components (DKG, tPKE, and zkSNARG), e.g., [GM24,ENP24,CLW25,PNSN26,GMNO18], and for their secure, efficient composition under a common assumption. Replacing DKG + tPKE with VSS + SKE avoids the difficulties of threshold public-key encryption, but leaves the first approach's underlying VSS and zkSNARG issues intact.

The third approach, considered in [AKP22], is computationally efficient as it runs the MPC only once on the dealer side. It also naturally supports post-quantum security as it builds on MPC and BA protocols that can be instantiated in an information-theoretically secure manner. Protocols known as Distributed Verifier Zero-Knowledge, e.g., [ACF02, YW22, EPSW24], can work similarly for sharing a witness. However, besides being inherently interactive among the parties, "third-party evidence" of correctness is less clean because validity is tied to the BA procedure (who voted what, when) rather than a single self-contained proof artifact. Besides, this approach does not prevent a dealer from carrying out *fraud* through collusion with $t+1$ recipients. Namely, it cannot rule out scenarios where the dealer induces reliance while never possessing the promised secret in the first place. On the other hand, our WS scheme works as a proof of knowledge, preventing such "empty promises," even if the dealer later refuses to reveal the secret. In a purely cryptographic sense, "not revealing later" and "never having had it" look the same as adversarial behaviors. However, it can be a crucial legal and practical distinction in the real life: In contract/tort doctrine, a misrepresentation of present knowledge can ground promissory fraud [Ame77], while fraudulent or material misrepresentation can make a contract voidable and trigger remedies that differ from ordinary breach [Ame81]. Modern scholarship likewise emphasizes that courts treat the contract–fraud boundary as a serious policy issue rather than a purely technical distinction [Zac24].

5 Offline Trapdoor-Extractable Trapdoor Commitments

5.1 Definitions

Trapdoor commitments are a type of commitment schemes where, if P knows a *trapdoor*, they can open a commitment to any message. Without the knowledge of the trapdoor, however, P can only open the commitment to reveal the originally committed message, preserving binding. In other words, the commitment is only binding if the trapdoor is not known to P. A trapdoor commitment scheme has four properties: *completeness*, *hiding*, *binding*, and *equivocability*. We formalize these properties below, following [CV05].

Definition 21 (Trapdoor Commitment Scheme). *A trapdoor commitment scheme is a tuple of five algorithms* $\mathcal{TD} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{Equiv}, \mathsf{Ver})$ *for a message space* $\mathcal{M}$ *such that:*

$\mathsf{Gen}(1^\lambda) \to (ck, td)$: *On input the security parameter* λ, $\mathsf{Gen}(1^\lambda)$ *returns a commitment key* ck *and a trapdoor* td.

$\mathsf{Com}(ck, m) \to (com, open)$: *On input a commitment key* ck *and a message* $m \in \mathcal{M}$, $\mathsf{Com}(ck, m)$ *outputs a commitment* com *and an opening* $open$.

$\mathsf{TCom}(ck, td) \to (com, st)$: *On input a commitment key* ck *and a trapdoor* td, $\mathsf{TCom}(ck, td)$ *outputs a commitment* com *and a state* st.

$\mathsf{Equiv}(m, st) \to open$: *On input a message* $m \in \mathcal{M}$ *and a state* st, $\mathsf{Equiv}(m, st)$ *outputs an opening* $open$.

$\mathsf{Ver}(ck, com, m, open) \to 0/1$: *On input a commitment key* ck, *a commitment* com, *a message* m, *and an opening* $open$, $\mathsf{Ver}(ck, com, m, open)$ *outputs 1 (accept) or 0 (reject).*

Moreover, these algorithms must satisfy the following properties:

Completeness. *For any* $m \in \mathcal{M}$,

$$\Pr\left[\mathsf{Ver}(ck, com, m, open) = 1 \;\middle|\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, open) \leftarrow \mathsf{Com}(ck, m)\end{array}\right] = 1$$

Binding. *A trapdoor commitment scheme is computationally binding if for any PPT adversary* $\mathcal{A}$, *the following advantage is* $\leq \mathsf{negl}(\lambda)$*:*

$$\mathsf{Adv}^{\mathsf{bin}}_{\mathcal{TD},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l}\mathsf{Ver}(ck, com, m, open) = 1\\ \wedge\, \mathsf{Ver}(ck, com, m', open') = 1\\ \wedge\, m \neq m'\end{array} \;\middle|\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, m, open, m',\\ open') \leftarrow \mathcal{A}(ck)\end{array}\right]$$

If the property holds for any (even computationally unbounded) adversary $\mathcal{A}$, *then the scheme is statistically binding.*

Hiding. *A trapdoor commitment is computationally (resp. statistically) hiding if for any PPT (resp. unbounded) adversary* $\mathcal{A}$, *and for every* $m, m' \in \mathcal{M}$,

$$\left|\Pr\left[1 \leftarrow \mathcal{A}(ck, com) \;\middle|\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, open) \leftarrow \mathsf{Com}(ck, m)\end{array}\right]\right.$$
$$\left. - \Pr\left[1 \leftarrow \mathcal{A}(ck, com) \;\middle|\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, open) \leftarrow \mathsf{Com}(ck, m')\end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

Equivocability. *A trapdoor commitment is computationally (resp. statistically) equivocable if for any* $m \in \mathcal{M}$ *the following distributions are computationally (resp. statistically) indistinguishable:*

$$\left\{(ck, com, open, m) \;:\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, open) \leftarrow \mathsf{Com}(ck, m)\end{array}\right\} \quad \textit{and}$$
$$\left\{(ck, com, open, m) \;:\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, st) \leftarrow \mathsf{TCom}(ck, td)\\ open \leftarrow \mathsf{Equiv}(st, m)\end{array}\right\}.$$

We introduce a property called (offline) trapdoor extractability, for which one can extract a valid trapdoor from any adversary that breaks binding. We require the extractor is "offline", meaning that it only gets as input the commitment and two valid openings, without oracle access to the adversary. This way, the extractor can be used in the real protocol. It is assumed that there exists a trapdoor-checking function f_t such that $f_t(ck, td) = 1$ if and only if td is a valid trapdoor for ck.

Definition 22 (Offline Trapdoor Extractability). *A trapdoor commitment is* offline trapdoor extractable *if there exists an offline strict polynomial-time extractor* $\mathsf{Ext}_{\mathsf{off}}$ *that, for any PPT adversary* $\mathcal{A}$,

$$\Pr\left[\begin{array}{l}f_t(ck, td') = 0\\ \wedge\, \mathsf{Ver}(ck, com, m, open) = 1\\ \wedge\, \mathsf{Ver}(ck, com, m', open') = 1\\ \wedge\, m \neq m'\end{array} \;\middle|\; \begin{array}{l}(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)\\ (com, m, open, m',\\ open') \leftarrow \mathcal{A}(ck)\\ td' \leftarrow \mathsf{Ext}_{\mathsf{off}}(ck, com, m, open, m',\\ open')\end{array}\right] \leq \mathsf{negl}(\lambda).$$

5.2 Construction from Multi-round ZK proof systems

Trapdoor Commitment from Critical-Round ZK [ABB+26]. We first recall the construction of trapdoor commitment for multi-round proof systems having a critical round ZK in [ABB+26]. Given a $(2\mu+1)$-move public-coin proof protocol, $\Pi_{i^*_{\mathsf{zk}}} = (\mathsf{A}, \mathsf{Z}, \mathsf{V}, \mathsf{SimA}, \mathsf{SimZ})$, for relation R that is CRZK at round i^*_{zk}, and hash functions $H_i : \{0,1\}^* \to \mathcal{C}_i$, the construction of trapdoor commitment scheme $\mathsf{K} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{Equiv}, \mathsf{Ver})$ is illustrated in Figure 3 with the message space $\mathcal{M} = \mathcal{C}_{i^*_{\mathsf{zk}}}$. Let Π be CRZK at round i^*_{zk} such that the simulator only needs to know the challenge $c_{i^*_{\mathsf{zk}}}$ in advance. Then, the trapdoor commitment scheme is constructed by fixing $m = c_{i^*_{\mathsf{zk}}}$ and using Fiat-Shamir to compute the remaining challenges. In more detail, given (x, m) as input, the committer:

- Fixes $c_{i^*_{\mathsf{zk}}} = m$ and runs the zero-knowledge simulator of Π to obtain a.
- Computes c_i as in (5) for $i \neq i^*_{\mathsf{zk}}$, where z_i is also obtained by running the simulator.
- The commitment is the transcript up to round i^*_{zk}: $com := (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$.
- The opening is the remaining part of the transcript, given by $open := (z_{i^*_{\mathsf{zk}}}, \ldots, z_\mu)$.

This trapdoor commitment satisfies hiding, equivocation, binding, and trapdoor extractability properties. The hiding and equivocation properties follow from CRZK at round i^*_{zk}: there exist two simulators $(\mathsf{SimA}, \mathsf{SimZ})$ that generate indistinguishable transcripts, where SimA only takes as input a commitment key and a message (as the challenge at round i^*_{zk}). Furthermore, it is binding and trapdoor extractability if $\Pi_{i^*_{\mathsf{zk}}}$ is (extended) $\vec{k}$-SS with $k_{i^*_{\mathsf{zk}}} = 2$. The proof for binding and (online) trapdoor extractability requires a rewinding technique to show that if there exists an adversary that breaks the binding with non-negligible probability, we can construct an adversary to either find a trapdoor for the corresponding commitment key or break the hardness assumption.

Theorem 6 (Restated from [ABB+26]). *$\mathsf{K}_{\Pi_{i^*_{\mathsf{zk}}}}$ in Figure 3 constitutes a trapdoor commitment scheme with the following properties:*

- *It is hiding and equivocal if $\Pi_{i^*_{\mathsf{zk}}}$ is CRSHVZK at round i^*_{zk}.*
- *It is binding if $\Pi_{i^*_{\mathsf{zk}}}$ is perfectly complete and (extended) $\vec{h}$-SS with $h_{i^*_{\mathsf{zk}}} = 2$, relations R_{hd} and R are one-way, H_i is a programmable random oracle, and $h_i/|\mathcal{C}_i|$ is negligible for every $i \neq i^*_{\mathsf{zk}}$.*

Offline trapdoor extractability from CRSS. Offline trapdoor extractability guarantees that a trapdoor can be extracted in *strict* polynomial time from just the output of a binding adversary, without needing to rewind it. Therefore, to achieve *offline* trapdoor extractability, we cannot rely on the proof technique in [ABB+26] anymore. This is where CRSS becomes crucial. It provides a critical extractor CrExt that runs in strict polynomial time given a small critical tree, and this is precisely the offline extractor we need. If CrExt fails to output a witness on a critical tree T^*, the extended extractor ExExt finds a solution to the hard problem from any accepting extended tree $T \sqsupseteq T^*$ satisfying $\boldsymbol{\Phi}(T) = 1$. Under the conditions below, Theorem 2 bounds this failure probability by constructing such a tree in the reduction.

Theorem 7. *Let $\Pi_{i^*_{\mathsf{zk}}}$ be a protocol stated in Theorem 6. Assume that*

- *$\Pi_{i^*_{\mathsf{zk}}}$ is extended $(\vec{h}, \boldsymbol{\Phi})$-SS and $(\vec{k}, \Phi_{cr})$-CRSS and $i^*_{\mathsf{ss}} = i^*_{\mathsf{zk}}$,*
- *$k_{i^*_{\mathsf{zk}}} = h_{i^*_{\mathsf{zk}}} = 2$ and $k_i = 1$ for all $i \neq i^*_{\mathsf{zk}}$; equivalently, i^*_{zk} is the only branching round in the critical tree,*
- *Φ_{cr} is trivial, $\boldsymbol{\Phi}$ does not depend on any challenge from round i^*_{zk}, and $\boldsymbol{\Phi}|_{\vec{k}}(T^*) = 1$ for every accepting critical tree T^*,*
- *R_{hd} and R are one-way, the H_i are programmable random oracles, and for every $i \neq i^*_{\mathsf{zk}}$, both $\frac{h_i}{|\mathcal{C}_i|}$ and $\sum_{\ell=1}^{h_i} p^{\mathsf{chal}}_{i,\ell}$ are negligible.*

Then $\mathsf{K}_{\Pi_{i^*_{\mathsf{zk}}}}$ in Figure 3 supports offline trapdoor extraction.

Proof. We rely on the CRSS property to complete our proof. First observe that two valid openings $(com, m, open)$ and $(com, m', open')$ with $m \neq m'$ form a $\vec{k}$-critical tree T^*, branching only at round i^*_{zk}. We let $\mathsf{Ext}_{\mathsf{off}} := \mathsf{CrExt}$.

Gen: Given 1^λ as input, compute $(x, w) \leftarrow\!\$\ \mathcal{L}_{RW}(\lambda)$. Commitment key is x, and w is a trapdoor.

Com: Given (x, m) as input, compute $(st_1, a) \leftarrow \mathsf{SimA}(x, m)$ and $(st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)$ for $i = 1, \ldots, \mu$ where

$$c_i := \begin{cases} H_i(x, a, z_1, \ldots, z_{i-1}) & (i < i^*_{\mathsf{zk}}) \\ m & (i = i^*_{\mathsf{zk}}) \\ H_i(x, a, m, z_1, \ldots, z_{i-1}) & (i > i^*_{\mathsf{zk}}). \end{cases} \tag{5}$$

Output $com := (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$ and $open := (z_{i^*_{\mathsf{zk}}}, \ldots, z_\mu)$.

TCom: Given (w, x) as input, compute $(st_1, a) \leftarrow \mathsf{A}(x, w)$ and $(st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i)$ for $i = 1, \ldots, i^*_{\mathsf{zk}} - 1$ for c_i as in (5).
Output $com := (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$ and $st_{i^*_{\mathsf{zk}}}$.

Equiv: Given $(st_{i^*_{\mathsf{zk}}}, m)$, compute $(st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i)$ for $i = i^*_{\mathsf{zk}}, \ldots, \mu$ for c_i following (5). Output $open := (z_{i^*_{\mathsf{zk}}}, \ldots, z_\mu)$.

Ver: Given x, com, m, and $open$ as input, parse them into $(x, a, z_1, \ldots, z_\mu)$. Compute each c_i as in (5) and output $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.

Fig. 3. Trapdoor commitment scheme $\mathsf{K}_{\Pi_{i^*_{\mathsf{zk}}}}$ [ABB+26].

Let

$$\epsilon_{\mathsf{off}} := \Pr\left[\begin{array}{l|l} f_t(ck, td') = 0 & (ck, td) \leftarrow \mathsf{Gen}(1^\lambda) \\ \wedge\ \mathsf{Ver}(ck, com, m, open) = 1 & (com, m, open, m', \\ \wedge\ \mathsf{Ver}(ck, com, m', open') = 1 & open') \leftarrow \mathcal{A}(ck) \\ \wedge\ m \neq m' & td' \leftarrow \mathsf{Ext}_{\mathsf{off}}(ck, com, m, open, m', \\ & open') \end{array}\right].$$

Let $\mathcal{A}'$ sample ck as in the experiment, run $\mathcal{A}(ck)$ and output (ck, T^*) if both openings verify and $m \neq m'$, and $\perp$ otherwise. Here $f_t(ck, td') = R(ck, td')$. The round-$i^*_{\mathsf{zk}}$ challenges are chosen by $\mathcal{A}$, while all other challenges follow (5). Since $k_{i^*_{\mathsf{zk}}} = h_{i^*_{\mathsf{zk}}} = 2$ and the restriction condition on $\boldsymbol{\Phi}$ holds, we have

$$\epsilon_{\mathsf{off}} \leq \kappa_{ex} + \epsilon_{hd}\theta,$$

from Theorem 2 where κ_{ex} and θ exclude round i^*_{zk}. Since $\theta \leq 1$ and both ϵ_{hd} and κ_{ex} are negligible under the stated conditions, ϵ_{off} is negligible.

□

Instantiation from KKW. We instantiate $\mathsf{K}_{\Pi_{i^*_{\mathsf{zk}}}}$ with the partial parallel repetition $\Pi_{\mathsf{PPR\text{-}KKW}}$ of Section 3.3, with $i^*_{\mathsf{zk}} = 2$, so that the committed message is the shared second-round challenge $c_2 \in [n]$. This answers in the affirmative the question left open in [ABB+26] of whether the KKW framework can be used to construct trapdoor commitments; see Section 1.2 for why the cut-and-choose variant of KKW does *not* qualify. The scheme commits to $\log_2 n$ bits. To support arbitrary message lengths, we pad the message to a length ℓ divisible by $\log_2 n$, split it into $\ell/(\log_2 n)$ chunks, and run $\ell/(\log_2 n)$ independent copies of the scheme, one per chunk.

Corollary 2. *Let $\Pi_{\mathsf{PPR\text{-}KKW}}$ be instantiated with a computationally binding commitment scheme and a collision-resistant hash function, let R be one-way, and let H_1 be a programmable random oracle. Then $\mathsf{K}_{\Pi_{\mathsf{PPR\text{-}KKW}}}$ in Figure 3 with $i^*_{\mathsf{zk}} = 2$ is a hiding, equivocal and binding trapdoor commitment scheme for message space $[n]$ that supports offline trapdoor extraction, with extraction error $2(Q+1)/m^\tau + \epsilon_{hd}$ for a Q-query adversary, where ϵ_{hd} bounds the probability of breaking binding or collision resistance.*

Proof. We verify the conditions of Theorem 7 with $\vec{h} = (2, 2)$ and $\vec{k} = (1, 2)$. By Lemma 1, $\Pi_{\mathsf{PPR\text{-}KKW}}$ is extended $(2, 2)$-SS with trivial $\boldsymbol{\Phi}$, plain $(1, 2)$-CRSS at round $i^*_{\mathsf{ss}} = 2$, and CRSHVZK at round $i^*_{\mathsf{zk}} = 2$; hence $i^*_{\mathsf{ss}} = i^*_{\mathsf{zk}} = 2$, and $k_{i^*_{\mathsf{zk}}} = h_{i^*_{\mathsf{zk}}} = 2$ with $k_1 = 1$, so round 2 is the only branching round of the critical tree. Since both predicate systems are trivial, $\boldsymbol{\Phi}$ does not depend on the round-2 challenge, and $\boldsymbol{\Phi}|_{\vec{k}}(T^*) = 1$ for every critical tree T^*. The hardness relation R_{hd}, consisting of hash collisions and double openings of the commitment scheme, is one-way by assumption, and R is one-way by assumption. Finally, the only round other than i^*_{zk} is round 1 with $\mathcal{C}_1 = [m]^\tau$, for which $h_1/|\mathcal{C}_1|$ is negligible. Hiding and equivocality follow from CRSHVZK at round i^*_{zk} and binding from extended

$(2,2)$-SS via Theorem 6, and Theorem 7 gives offline trapdoor extraction. The extraction error is the bound of Theorem 2 with $\kappa_{ex} = 2(Q+1)\max(1/m^{\tau}, 0) = 2(Q+1)/m^{\tau}$ and $\theta \leq 1$. □

6 A CRSS Variant of the Multi-Round Fischlin Transform

In this section, we sketch an application of plain CRSS to the multi-round straight-line Fischlin transform of [RT25]. For simplicity, we omit Φ_{cr} and $\boldsymbol{\Phi}$ in the following discussion, since they are not needed to convey the essential ideas.

The multi-round Fischlin transform [RT25]. The transform of [RT25] compiles a μ-round public-coin protocol Π which is $(k_1, \ldots, k_\mu)$-special sound into a non-interactive argument with *straight-line* knowledge extraction in the random oracle model, generalizing Fischlin's original transform [Fis05] for Σ-protocols. Roughly, the prover is asked to obtain a hash-based proof of work by outputting a transcript π such that $H(\pi)$ starts with ℓ leading zeros (See Figure 4 for details). To ensure that an extractor given as input the prover's random oracle queries can actually recover an $(k_1, \ldots, k_\mu)$-tree of accepting transcripts, the prover is forced to "probe" each round: when computing the round-i challenge by hashing the transcript prefix as in Fiat-Shamir, the prover iterates over an index set of size $k'_i = c \cdot k_i$ (for some small c, by the operator $\mathsf{increment}_{\mathbf{k}'}$), each index selecting a challenge from the full challenge space, and computes the prover responses for each. On input $\mathbf{j} = (j_1, \ldots, j_\mu) \in [k'_1] \times [k'_2] \times \cdots \times [k'_\mu]$, the operator $\mathsf{increment}_{\mathbf{k}'}$ increments $\mathbf{j}'$ as follows:

1. Find the last position $i \in [\mu]$ for which $j_i < k'_i$. If no such position exists, return $\perp$.
2. Return $\mathbf{j}' = (j_1, \ldots, j_{i-1}, j_i + 1, 1, \ldots, 1)$.

Input: Prover of underlying protocol P, operator $\mathsf{increment}_{\mathbf{k}'}$
Construction:

- Sample random coins $r_1, \ldots, r_t \leftarrow\!\!\$\ \{0,1\}^{\rho}$ for t executions of P.
- For $i = 1, \ldots, t$, invoke P on input (x, w) and randomness r_i to obtain a first messages $a_{1,i}$. Denote $\mathbf{a_1} = (a_{1,1}, \ldots, a_{1,t})$.
- For $i = 1, \ldots, t$:
  - Initialize $\mathbf{j_i} \leftarrow (1, 1, \ldots, 1) \in \mathbb{N}^{\mu}$ and $\mathsf{flag}_i \leftarrow \mathsf{false}$.
  - While $\mathsf{flag}_i = \mathsf{false}$ do:
    - Invoke P on (x, w), challenge indices $\mathbf{j_i}$, and randomness r_i to obtain a proof π_i.
    - If $H(\mathbf{a_1}, i, x, \pi_i) = 0^{\ell}$, then set $\mathsf{flag}_i = \mathsf{true}$. Otherwise, increment $\mathbf{j_i}$ by setting $\mathbf{j_i} \leftarrow \mathsf{increment}_{\mathbf{k}'}(\mathbf{j_i})$.
    - If $\mathbf{j_i} = \perp$, then output $\perp$ and abort.
- Output $\pi \leftarrow (\pi_1, \ldots, \pi_t, \mathbf{j}_1, \ldots, \mathbf{j}_t)$.

Fig. 4. Prover P' in the multi-round Fischlin transform [RT25].

A central lemma of [RT25] (Lemma 2) bounds the maximum number of leaves of the partial tree that a malicious prover can induce with q random oracle queries without enabling extraction. This bound takes the form of a sum over rounds $i \in [\mu]$ of terms depending on k'_i and k_i. With $k'_i = c \cdot k_i$, the prover tries at most $K = c^{\mu} \prod_i k_i$ index vectors per repetition. For $c = \Theta(\mu \log \lambda)$ and polynomially bounded $\prod_i k_i$, the condition $\mu = O(\log \lambda / \log\log \lambda)$ keeps this prover cost polynomial.

The CRSS optimization. The proof and challenge-probing strategy of [RT25] only needs a branching of size k_i at round i *if witness extraction itself needs* k_i *branches* at that round. For protocols that are $(\vec{k})$-CRSS at critical round i^*_{ss}, this by definition can be simplified for rounds $i < i^*_{\mathsf{ss}}$, where $k_i = 1$. The critical-round extractor CrExt reconstructs a witness from a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-critical subtree, in which the early rounds never branch. On such rounds, no probing is needed: the prover can simply use the standard Fiat-Shamir challenge $c_i := F_i(x, \tau_i, 1)$, where τ_i is the transcript prefix before challenge c_i.

Concretely, we use the prover, verifier and operator $\mathsf{increment}_{\mathbf{k}'}$ of [RT25] with

$$k'_i = \begin{cases} 1 & \text{if } i < i^*_{\mathsf{ss}}, \\ c \cdot k_i & \text{if } i \geq i^*_{\mathsf{ss}}, \end{cases}$$

where k_i is the critical-tree parameter of the underlying protocol. The challenges are $c_i = F_i(x, \tau_i, j_i)$ for $j_i \in [k'_i]$, where F_i maps into the full challenge space $\mathcal{C}_i$. The verifier checks these indices and challenges, the accepting transcripts and the proof of work as in [RT25]. During the extraction, the straight-line extractor only needs to rebuild the critical subtree from round i^*_{ss}, with its common early-round prefix. As a sanity check, the degenerate case $i^*_{\mathsf{ss}} = \mu + 1$ allowed by Definition 14 gives $k'_i = 1$ at every round and $L = 0$: no proof of work is needed and the transform reduces to plain Fiat–Shamir, which is why Theorem 8 below restricts to $i^*_{\mathsf{ss}} \in [\mu]$.

Improvement on the bound of round complexity and prover running time. The above replacement is precisely what is needed to start the summation in Lemma 2 of [RT25] at $i = i^*_{\mathsf{ss}}$ rather than at $i = 1$. Their induction also applies when $k_i = 1$. There are at most $k_i - 1 = 0$ heavy children, so the corresponding summand vanishes, and $k'_i = k_i = 1$ contributes the factor 1. The bound is applied separately to each tree with a fixed vector of first messages and a fixed repetition, as in [RT25]. The same induction step applies to any round with $k_i = 1$, not only to those before i^*_{ss}; we restrict the modification to the prefix $i < i^*_{\mathsf{ss}}$ for simplicity of presentation.

Let $m = \mu - i^*_{\mathsf{ss}} + 1$ and $K' = \prod_i k'_i = c^m \prod_{i \geq i^*_{\mathsf{ss}}} k_i$. As in [RT25, Sect. 3.5], $L \leq mK'/c$. Choosing ℓ such that

$$\ln(2t) \leq 2^{-\ell} K' \leq c/(2m)$$

gives $Q_{P^*}(2^{-\ell}L)^t \leq Q_{P^*} 2^{-t}$ and completeness error at most $1/2$, apart from the underlying completeness and challenge-collision errors below. For $t = \lambda$, we can take $c = \Theta(m \log \lambda)$ with a sufficiently large constant. Assuming polynomially bounded $\prod_i k_i$, the round-complexity constraint $\mu = \mathcal{O}(\log \lambda / \log\log \lambda)$ of [RT25] is then relaxed to

$$\mu - i^*_{\mathsf{ss}} + 1 = \mathcal{O}(\log \lambda / \log\log \lambda),$$

i.e., the constraint applies only to the number of rounds from the critical round. In addition, for protocols where i^*_{ss} is closer to μ, i.e., the number of remaining rounds from the critical round is only constant, the round-complexity constraint vanishes.

The prover tries at most K' index vectors per repetition, compared with $c^\mu \prod_i h_i$ when applying the original transform to the extended special-soundness parameter $\vec{h}$. After caching the prefix, rounds $i < i^*_{\mathsf{ss}}$ are computed once per repetition.

We then sketch the idea of security proof for our modified protocol. The argument of Lemma 1 of [RT25], with K' index vectors, gives completeness error at most $t(e^{-2^{-\ell}K'} + K'\epsilon_{\Pi}) + \eta$, where ϵ_{Π} is the completeness error of Π and η is the probability that two indices at the same prefix give the same challenge. A union bound gives $\eta \leq t \sum_i (\prod_{j<i} k'_j) \binom{k'_i}{2} / |\mathcal{C}_i|$. Straight-line extractability comes from the existence of a critical extractor CrExt. The extractor runs the prover once, records its queries and applies CrExt to an accepting critical tree. For knowledge soundness, we combine the tree-counting argument in the proof of Theorem 1 of [RT25] with the independent indexed tree-extension lemma below (Lemma 2). The reduction to R_{hd} uses rewinding and reprograms F_i including at rounds $i < i^*_{\mathsf{ss}}$. The straight-line extractor itself runs in strict polynomial time.

We use random-challenge unique responses in the sense of [RT25, Definition 5 and Fig. 1]: after fixing its statement, the adversary interacts with the random-challenge verifier with rewinding and wins with two recorded accepting transcripts whose first differing message is a prover response after the initial one. For protocols with generated public parameters, these are sampled honestly before the statement is fixed. All probabilities and advantages below include the same setup experiment as in Theorem 2.

Theorem 8. *Let (P, V) be a $(2\mu + 1)$-move critical-round $\vec{k}$-special sound and extended $\vec{h}$-special sound argument system at round $i^*_{\mathsf{ss}} \in [\mu]$ for relation $\mathcal{R}$ with hardness assumption on R_{hd}. Assume $k_i \leq h_i \leq |\mathcal{C}_i|$, polynomially bounded $\prod_i h_i$ and $\prod_i k'_i$. Let (P', V') be the non-interactive random oracle argument by the modified transformation with parameter $(t, \vec{k}', \ell)$, where $H, F_1, \ldots, F_\mu$ are independent random oracles. Then, there exists a straight-line extractor E such that for every x, and*

every PPT prover algorithm P^* there are algorithms A, B such that

$$\Pr\left[(x, E^{P^*}(x)) \in \mathcal{R}\right] \geq \epsilon_{P^*}(x) - \mathsf{Adv}^{rur}_{A,(P,V)}(\lambda) - Q_{P^*} \cdot (2^{-\ell} \cdot L)^t - \kappa_{ex} - \epsilon_{hd}\theta,$$

where

$$\epsilon_{P^*}(x) = \Pr\left[V'^{O}(x, \pi) = 1 : \pi \leftarrow\!\$\ P^{*O}(x)\right].$$

The query bound Q_{P^*} counts queries to $H, F_1, \ldots, F_\mu$, including prefix checks on challenge queries, transcript checks and the output-proof check. The advantage $\mathsf{Adv}^{rur}_{A,(P,V)}(\lambda)$ is as in random-challenge unique response security game in Definition 5 of [RT25], $L = \sum_{i=i^*_{\mathsf{ss}}}^{\mu}(k_i-1)\cdot\prod_{j=i+1}^{\mu} k'_j\cdot\prod_{l=1}^{i-1}(k'_l-k_l+1)$, κ_{ex} denotes $2(Q_{P^*}+1)\sum_{i=1}^{\mu}\frac{h_i-1}{|\mathcal{C}_i|}$, and $\theta = \prod_{i=1}^{\mu}\left(1-\frac{h_i-1}{|\mathcal{C}_i|}\right)$. Here ϵ_{hd} denotes the upper bound of the probability that B solves R_{hd}. Moreover, E and A run in polynomial time, and B runs in expected polynomial time. The CRSS optimization reduces L and the prover cost. κ_{ex} and θ still range over all rounds.

Lemma 2 (Indexed tree extension). *Let Π be a public-coin protocol with efficiently uniformly samplable challenge spaces $\mathcal{C}_i$ of known size N_i, with canonical representations of polynomial length, and let $1 \leq k_i \leq h_i \leq N_i$ for $i \in [\mu]$, with $M_h := \prod_i h_i$ polynomially bounded. Let $F_1, \ldots, F_\mu$ be independent classical random oracles, where $F_i(x, \tau_i, j)$ is uniform in $\mathcal{C}_i$ and τ_i is the full transcript prefix before challenge i. Let $\mathcal{A}$ be a PPT algorithm making at most Q oracle queries and outputting either $\perp$ or a statement x and an ordered accepting $\vec{k}$-tree T. Every edge at round i is annotated with an index j, its challenge is $F_i(x, \tau_i, j)$, and $\mathcal{A}$ queries every such input. Sibling challenges are pairwise distinct. Access to additional independent random oracles is allowed and included in Q.*

Write $\delta := \Pr[\mathcal{A}$ outputs $(x, T)]$ and $S := \sum_i (h_i - 1)/N_i$. There is a tree builder which, after an initial run of $\mathcal{A}$, outputs an accepting $\vec{h}$-tree $T \sqsupseteq T$ with probability at least $\delta - 2QS$. Its expected number of invocations of $\mathcal{A}$, including the initial one, is at most $M_h + Q(M_h - 1)$. It runs in expected polynomial time. The output is a tree for the underlying interactive protocol; it need not be consistent with a single random-oracle table.

Proof. Fix the public input and the coins of $\mathcal{A}$, and view the oracles as a random table J. At each level, use a random ordering of the challenge space independent of J and of the orderings at the other levels. Reset these orderings on each recursive call; they can be generated lazily. All oracle modifications made by a call are restored on return. The bounds below are uniform in the public input and the fixed coins, so we average over them at the end.

At a round-i node, first attempt to complete all k_i supplied children recursively, retaining their original order. If one fails, mark this node as failed, but still attempt the other supplied subtrees for the cost accounting below. If all succeed and $h_i > k_i$, let $q = (x, \tau_i, j)$ be the input of the first supplied child. Visit the challenges in the level-i random ordering, excluding the k_i supplied challenges. For each candidate b, rerun $\mathcal{A}$ with $J(q)$ replaced by b. Keep a candidate only if the new output contains the same node with q as its first child's input and that child completes recursively. Append the completed child, and stop after $h_i - k_i$ additions. Fail if the challenge space is exhausted. Because q fixes the statement and prefix, each retained child has the required parent. This constructs an accepting extension and preserves the leftmost containment convention.

Fix a round-i input q, all entries of J except $J(q)$, and all lower-level orderings. Let G_q be the set of answers b for which the rerun outputs a tree with q as the first input at that node and that first child completes. Set $a_q := |G_q|$. This set is independent of the original value $J(q)$. If $a_q > 0$, the initial run queries q: whether a fixed-coin oracle algorithm reaches its first query to q is independent of the answer at q. The output-query convention is used here. Consequently,

$$\sum_q \Pr[a_q > 0] \leq Q.$$

If an initial output has all supplied children completed at this node, then $J(q) \in G_q$, an event of probability at most a_q/N_i. For $a_q \geq h_i$, excluding the k_i supplied challenges leaves at least $h_i - k_i$ successful candidates, so completion at this node cannot fail. For $a_q < h_i$, the probability of an initial success followed by failure here is at most $(h_i - 1)/N_i$. No additional failure is possible when $h_i = k_i$.

For clarity, define p_i as the probability that the initial output exists and every subtree rooted at an original round-i node completes, using the same lower-level orderings in these events. Set $p_{\mu+1} = \delta$.

The children of the original round-i nodes are exactly the roots of the original level-$(i+1)$ forest. Thus p_{i+1} supplies the event that all preserved children complete. The preceding local bound and query-support inequality give

$$p_i \geq p_{i+1} - Q\frac{h_i - 1}{N_i}.$$

Completion of every new candidate's entire lower subtree is already included in G_q; no independent success assumption about those subtrees is used. Telescoping yields $p_1 \geq \delta - QS \geq \delta - 2QS$.

It remains to bound the time spent on unsuccessful candidates and recursive completions. Write $a = a_q$, $h = h_i$, $k = k_i$, and $N = N_i$. For $a \geq h > k$, a fixed candidate in G_q can be visited only when it differs from the initial answer; conditional on eligibility, its probability of being among the first $h-k$ remaining successes is at most $(h-k)/(a-k)$. For a candidate outside G_q, its probability of preceding the $(h-k)$th remaining success is at most $(h-k)/(a-k+1)$. Averaging over the initial answer, its visitation probability is therefore at most

$$\frac{1}{N}\max\left\{\frac{(a-1)(h-k)}{a-k}, \frac{a(h-k)}{a-k+1}\right\} \leq \frac{h-1}{N}.$$

The exclusion set may depend on the initial answer; only its size k was used. For $a < h$, the bound follows from $a/N \leq (h-1)/N$. For $h = k$, no candidates are visited. Together with query support, this bounds the expected number of additional invocations at level i by $(h_i - 1)Q$.

Let C_i be the expected number of invocations needed to attempt completion of the entire original level-i forest, counting the initial invocation once. For a table J, let $\gamma_q(J)$ be the additional invocation cost of completing the first child at input q, and zero when that child is absent from the output. For every fixed table and lower-level orderings, these first-child subtrees are a subset of the supplied level-$(i+1)$ forest. Since every supplied subtree is attempted, even if another fails,

$$\mathbb{E}\left[\sum_q \gamma_q(J)\right] \leq C_{i+1} - 1.$$

The visitation bound holds for each fixed candidate and lower-level orderings. Multiplying it by that candidate's completion cost, summing over its values and then over q, bounds the expected additional recursive cost by $(h_i - 1)(C_{i+1} - 1)$. It follows that

$$C_i \leq h_i C_{i+1} + (h_i - 1)Q, \qquad C_{\mu+1} = 1,$$

and hence $C_1 \leq M_h + Q(M_h - 1)$. To implement the random orderings, draw independent uniform challenges, cache each previously unseen value, and discard duplicates. Reset only the cursor on a recursive call, reusing the same cached ordering at that level. Generating its first r distinct values takes expected

$$N_i \sum_{s=0}^{r-1} \frac{1}{N_i - s} \leq r(1 + \ln N_i)$$

uniform draws. The waiting times are independent of the resulting ordering, so the same bound applies when r is determined by that ordering and the oracle table. Each traversal skips at most k_i supplied challenges in addition to the candidates counted above. Using a dictionary for the cache gives polynomial overhead in the number and representation length of visited values. Together with polynomial transcript size, this proves the expected-time claim. □

Corollary 3 (Completing both transcripts). *Let A issue at most Q queries as in Lemma 2 and, on an efficiently testable event of probability δ, output two accepting transcripts with query annotations. For a target shape $\vec{h}$ satisfying that lemma, both transcripts can be preserved and completed to accepting $\vec{h}$-trees with joint probability at least*

$$\delta - 4Q\sum_i \frac{h_i - 1}{N_i}.$$

The expected number of invocations of A is at most $2\big(M_h + Q(M_h - 1)\big)$, where $M_h = \prod_i h_i$.

Proof. For $b \in \{0,1\}$, let A_b run A and output transcript b with its annotations if the event holds, and $\perp$ otherwise. Apply Lemma 2 with $\vec{k} = (1,\dots,1)$ to each A_b, using the same initial oracle table and coins of A, and restoring all reprogramming between the two builders. Both initial runs then select the same pair. Each builder fails to complete its preserved transcript on the initial event with probability at most $2Q\sum_i (h_i - 1)/N_i$. A union bound gives the joint guarantee; no independence of the two completion events is needed. Adding the expected costs gives the stated bound. The resulting trees are interactive transcript trees and need not share a final oracle table. □

Applying the corollary to RUR. To use Corollary 3 in the RUR game of [RT25, Definition 5 and Fig. 1], simulate its Ver oracle and maintain its sets of partial and complete transcripts. Reject an unrecorded nonempty prefix. For each valid call requiring a round-i challenge, use $F_i(x, \tau_i, j)$ with a fresh invocation index j, and record the resulting partial transcript and its annotation. Thus repeated calls at the same prefix receive fresh independent challenges, exactly as in that game. Reusing a recorded prefix does not resample its earlier challenges. At the final round, check acceptance and record the complete transcript. The simulator tests that the output pair belongs to the recorded complete transcripts and meets the RUR winning condition. This is an exact simulation, using at most one random-oracle query per Ver call. In particular, the following example proofs use RT25's RUR definition itself; indices only record the random samples used inside the reduction.

Proof (Proof of Theorem 8.). Let acc denote acceptance of the output proof and tree the event that the recorded queries contain an accepting critical tree T^*. We use the response-collision reduction of [RT25, proof of Theorem 1], making its verifier simulation explicit. Fix the target statement x before the oracle interaction. Before answering a fresh query $F_i(x, \tau_i, j)$, check each earlier challenge in τ_i against its entire allowed indexed set, recursively sampling missing entries and caching every answer. The recursion only calls earlier rounds and uses polynomially many queries, since $\prod_i k'_i$ is polynomially bounded. Count these additional queries in Q_{P^*}.

If the checks pass, the preceding verifier prefix is already recorded in RT25's game. Answer the fresh query by calling Ver with that prefix and the next prover message. For an invalid prefix, or a different statement, sample and cache an independent answer locally. An invalid prefix cannot become valid later: a failed membership check has already sampled its entire allowed challenge set. Repeated oracle inputs reuse cached answers, whereas different indices at a valid prefix use fresh Ver calls. This exactly simulates the independent random oracles, including queries made out of transcript order. Register each checked accepting transcript by submitting its final response to Ver. Thus any two such transcripts that first differ at a prover response give a win in RT25's RUR game. This collision reduction does not use special soundness.

Group the checked accepting transcripts by their first-message vector $\mathbf{a}_1$ and repetition $r \in [t]$. Outside the collision event, each round-i node has at most k'_i children with distinct challenges. If no critical tree exists, each such tree has at most L leaves by their Lemma 2 and the non-branching case established above.

For the adaptive probability calculation, assign to each new first-message vector t independent streams of fresh uniform H-answers, one for each repetition. Check transcript validity and indexed-challenge membership before answering its fresh proof-of-work query; include these checks and final verification in Q_{P^*}. If the final proof is accepted without a critical tree or a response collision, its vector must have a zero among the first L answers of each stream. Each vector has probability at most $(2^{-\ell}L)^t$ of this event, conditional on its creation history, and at most Q_{P^*} vectors are created. Hence

$$\Pr[\mathsf{acc} \wedge \neg\mathsf{tree}] \le \mathsf{Adv}^{\mathrm{rur}}_{A,(P,V)}(\lambda) + Q_{P^*}(2^{-\ell}L)^t.$$

Fix the CRSS extractors and a deterministic rule for selecting the critical tree. Let $\mathcal{A}'$ run the prover and query-record extractor, outputting the selected accepting T^* with its query annotations only if $R(x, \mathsf{CrExt}(x, T^*)) = 0$. Let δ be this output probability. Apply Lemma 2 with $Q = Q_{P^*}$ and $S = \sum_i (h_i - 1)/|\mathcal{C}_i|$. The resulting builder preserves T^* and completes an accepting $\vec{h}$-tree with probability at least $\delta - 2Q_{P^*}S$, in expected polynomial time. By plain CRSS, every successful completion gives a solution to R_{hd} through ExExt. Thus the resulting reduction B satisfies $\delta - 2Q_{P^*}S \le \epsilon_{hd}$ in the same hardness experiment. This application permits branching after i^*_{ss} and $k_{i^*_{\mathsf{ss}}} < h_{i^*_{\mathsf{ss}}}$; it does not invoke the restricted tree shape in Theorem 2.

Finally, $1 - \theta \le S$, $\delta \le 1$ and $\kappa_{ex} = 2(Q_{P^*} + 1)S$ imply

$$\theta(\delta - 2Q_{P^*}S) - (\delta - \kappa_{ex}) = 2S - (1 - \theta)(\delta - 2Q_{P^*}S) \ge 0.$$

Consequently $\delta \leq \kappa_{ex} + \epsilon_{hd}\theta$. Combining the two failure events proves the claim. Only the hardness reduction rewinds; the straight-line extractor runs the prover once and applies CrExt to its recorded queries. □

7 Conclusion

This paper revisits the soundness mechanism of multi-round public-coin proof systems as a real executable component in cryptographic protocols, aiming to narrow the practical gap to three-move protocols. To this end, we introduced Critical-Round Special Soundness, which enables witness extraction from only a small subtree of the usual transcript tree. This recovers the "extractor usability" advantages of 3-move protocols for multi-round proofs, yielding applications such as witness sharing, trapdoor commitments and better bounds for straight-line extraction transforms.

While efficiency can be unfavorable when the presented applications are built from other post-quantum secure assumptions, our MPCitH-based construction provides a meaningful point in the hash-based "minicrypt" design space where hash functions alone suffice. Improving the concrete efficiency of CRSS instantiations is an open problem that will further increase the utility of multi-round public-coin proofs as reusable protocol building blocks.

Acknowledgements

This work was partially done while Dung Bui and David Balbás were doing an internship at NTT Social Informatics Laboratories, while David Balbás was at IMDEA Software Institute, Dung Bui was at Sorbonne University and Zehua Shang was at Kyoto University.
David Balbás was supported by the PICOCRYPT project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101001283), partially supported by projects PRODIGY (TED2021-132464B-I00) and ESPADA (PID2022-142290OB-I00) funded by MCIN/AEI/10.13039/501100011033/ and the European Union NextGenerationEU / PRTR.
Generative AI tools (Anthropic's Claude Fable 5.1 and OpenAI's ChatGPT Astra) were used for self-review and to assist with revisions to improve the overall quality of the paper. They were also used to generate some of the code used in the experiments. Their outputs were checked and edited by the authors.
Zhiyu Peng's participation in the early stages of this work is appreciated. We also thank Miguel Ambrona for useful comments about Plonk, and Marius Aardal and Sebastian Kolby for helpful discussions regarding the predicate special soundness framework of [AAB+24].
This paper was prepared in part for information purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co and its affiliates ("JP Morgan"), and is not a product of the Research Department of JP Morgan. JP Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

References

AAB+21. Masayuki Abe, Miguel Ambrona, Andrej Bogdanov, Miyako Ohkubo, and Alon Rosen. Acyclicity programming for sigma-protocols. In Kobbi Nissim and Brent Waters, editors, *TCC 2021: 19th Theory of Cryptography Conference, Part I*, volume 13042 of *Lecture Notes in Computer Science*, pages 435–465, Raleigh, NC, USA, November 8–11, 2021. Springer, Cham, Switzerland.

AAB+24. Marius A. Aardal, Diego F. Aranha, Katharina Boudgoust, Sebastian Kolby, and Akira Takahashi. Aggregating falcon signatures with LaBRADOR. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part I*, volume 14920 of *Lecture Notes in Computer Science*, pages 71–106, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

ABB+26. Masayuki Abe, David Balbás, Dung Bui, Miyako Ohkubo, Zehua Shang, Akira Takahashi, and Mehdi Tibouchi. Critical rounds in multi-round proofs: Proof of partial knowledge and trapdoor commitments. In Joan Daemen and Emmanuel Thomé, editors, *Advances in Cryptology – EUROCRYPT 2026, Part VII*, volume 16547 of *Lecture Notes in Computer Science*, pages 359–388, Rome, Italy, May 10–14, 2026. Springer, Cham, Switzerland.

ABO+24. Masayuki Abe, Andrej Bogdanov, Miyako Ohkubo, Alon Rosen, Zehua Shang, and Mehdi Tibouchi. CDS composition of multi-round protocols. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part IX*, volume 14928 of *Lecture Notes in Computer Science*, pages 391–423, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

ACF02. Masayuki Abe, Ronald Cramer, and Serge Fehr. Non-interactive distributed-verifier proofs and proving relations among commitments. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 206–223, Queenstown, New Zealand, December 1–5, 2002. Springer Berlin Heidelberg, Germany.

ACK21. Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed Σ-protocol theory for lattices. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 549–579, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland.

AF22. Thomas Attema and Serge Fehr. Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 415–443, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

AFK23. Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-Shamir transformation of multi-round interactive proofs (extended version). *Journal of Cryptology*, 36(4):36, October 2023.

AFR23. Thomas Attema, Serge Fehr, and Nicolas Resch. Generalized special-sound interactive proofs and their knowledge soundness. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023: 21st Theory of Cryptography Conference, Part III*, volume 14371 of *Lecture Notes in Computer Science*, pages 424–454, Taipei, Taiwan, November 29 – December 2, 2023. Springer, Cham, Switzerland.

AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 2087–2104, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press.

AK25. Benny Applebaum and Eliran Kachlon. How to share an NP statement or combiners for zero-knowledge proofs. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 480–513, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

AKLY24. Thomas Attema, Michael Klooß, Russell W. F. Lai, and Pavlo Yatsyna. Adaptive special soundness: Improved knowledge extraction by adaptive useful challenge sampling. Cryptology ePrint Archive, Report 2024/2038, 2024.

AKP22. Benny Applebaum, Eliran Kachlon, and Arpita Patra. Verifiable relation sharing and multi-verifier zero-knowledge in two rounds: Trading NIZKs with honest majority - (extended abstract). In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 33–56, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

Ame77. American Law Institute. *Restatement of the Law, Second: Torts*, volume 3. American Law Institute, 1977. Misrepresentation, including §530 (Misrepresentation of Intention).

Ame81. American Law Institute. *Restatement of the Law, Second: Contracts*. American Law Institute Publishers, 1981. Including §164 (When a Misrepresentation Makes a Contract Voidable).

BBB+18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334, San Francisco, CA, USA, May 21–23, 2018. IEEE Computer Society Press.

BBC+19. Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear pcps. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, Lecture Notes in Computer Science, pages 67–97. Springer, 2019.

BC23. Benedikt Bünz and Binyi Chen. Protostar: Generic efficient accumulation/folding for special-sound protocols. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part II*, volume 14439 of *Lecture Notes in Computer Science*, pages 77–110, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.

Beu20. Ward Beullens. Sigma protocols for MQ, PKP and SIS, and Fishy signature schemes. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 183–211, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland.

BF23. Benedikt Bünz and Ben Fisch. Multilinear schwartz-zippel mod N and lattice-based succinct arguments. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023: 21st Theory of Cryptography Conference, Part III*, volume 14371 of *Lecture Notes in Computer Science*, pages 394–423, Taipei, Taiwan, November 29 – December 2, 2023. Springer, Cham, Switzerland.

BGI16. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 1292–1303, Vienna, Austria, October 24–28, 2016. ACM Press.

BGKM23. Loïc Bidoux, Philippe Gaborit, Mukul Kulkarni, and Víctor Mateu. Code-based signatures from new proofs of knowledge for the syndrome decoding problem. *Designs, Codes and Cryptography*, 91(2):497–544, 2023.

BS23. Ward Beullens and Gregor Seiler. LaBRADOR: Compact proofs for R1CS from module-SIS. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part V*, volume 14085 of *Lecture Notes in Computer Science*, pages 518–548, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

CDM00. Ronald Cramer, Ivan Damgård, and Philip D. MacKenzie. Efficient zero-knowledge proofs of knowledge without intractability assumptions. In Hideki Imai and Yuliang Zheng, editors, *PKC 2000: 3rd International Workshop on Theory and Practice in Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 354–372, Melbourne, Victoria, Australia, January 18–20, 2000. Springer Berlin Heidelberg, Germany.

CDS94. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187, Santa Barbara, CA, USA, August 21–25, 1994. Springer Berlin Heidelberg, Germany.

CGM16. Melissa Chase, Chaya Ganesh, and Payman Mohassel. Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 499–530, Santa Barbara, CA, USA, August 14–18, 2016. Springer Berlin Heidelberg, Germany.

CLW25. Valerio Cini, Russell W. F. Lai, and Ivy K. Y. Woo. Pilvi: Lattice threshold PKE with small decryption shares and improved security. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology – ASIACRYPT 2025, Part VI*, volume 16250 of *Lecture Notes in Computer Science*, pages 539–569, Melbourne, VIC, Australia, December 8–12, 2025. Springer, Singapore, Singapore.

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

CV05. Dario Catalano and Ivan Visconti. Hybrid trapdoor commitments and their applications. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP 2005: 32nd International Colloquium on Automata, Languages and Programming*, volume 3580 of *Lecture Notes in Computer Science*, pages 298–310, Lisbon, Portugal, July 11–15, 2005. Springer Berlin Heidelberg, Germany.

Dam89. Ivan Damgård. On the existence of bit commitment schemes and zero-knowledge proofs. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, Lecture Notes in Computer Science, pages 17–27. Springer, 1989.

DFMS22. Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Online-extractability in the quantum random-oracle model. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 677–706, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland.

DG16. Massoud Hadian Dehkordi and Reza Ghasemi. A lightweight public verifiable multi secret sharing scheme using short integer solution. *Wirel. Pers. Commun.*, 91(3):1459–1469, 2016.

DG23. Quang Dao and Paul Grubbs. Spartan and bulletproofs are simulation-extractable (for free!). In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part II*, volume 14005 of *Lecture Notes in Computer Science*, pages 531–562, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

EBM12. Rachid El Bansarkhani and Mohammed Meziani. An efficient lattice-based secret sharing construction. In Ioannis Askoxylakis, Henrich C. Pöhls, and Joachim Posegga, editors, *Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems*, pages 160–168. Springer Berlin Heidelberg, 2012.

EG23. Liam Eagen and Ariel Gabizon. ProtoGalaxy: Efficient ProtoStar-style folding of multiple instances. Cryptology ePrint Archive, Report 2023/1106, 2023.

ENP24. Thomas Espitau, Guilhem Niot, and Thomas Prest. Flood and submerse: Distributed key generation and robust threshold signature from lattices. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part VII*, volume 14926 of *Lecture Notes in Computer Science*, pages 425–458, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

EPSW24. Daniel Escudero, Antigoni Polychroniadou, Yifan Song, and Chenkai Weng. Multi-verifier zero-knowledge proofs for any constant fraction of corrupted verifiers. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024: 31st Conference on Computer and Communications Security*, pages 4092–4106, Salt Lake City, UT, USA, October 14–18, 2024. ACM Press.

Fis01. Marc Fischlin. Trapdoor commitment schemes and their applications, 01 2001.

Fis05. Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 152–168, Santa Barbara, CA, USA, August 14–18, 2005. Springer Berlin Heidelberg, Germany.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer Berlin Heidelberg, Germany.

GJS19. Vipul Goyal, Aayush Jain, and Amit Sahai. Simultaneous amplification: The case of non-interactive zero-knowledge. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 608–637, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

GKK+22. Chaya Ganesh, Hamidreza Khoshakhlagh, Markulf Kohlweiss, Anca Nitulescu, and Michal Zajac. What makes Fiat-Shamir zkSNARKs (updatable SRS) simulation extractable? In Clemente Galdi and Stanislaw Jarecki, editors, *SCN 22: 13th International Conference on Security in Communication Networks*, volume 13409 of *Lecture Notes in Computer Science*, pages 735–760, Amalfi, Italy, September 12–14, 2022. Springer, Cham, Switzerland.

GM24. Matthias Geihs and Hart Montgomery. LaKey: Efficient lattice-based distributed PRFs enable scalable distributed key management. In Davide Balzarotti and Wenyuan Xu, editors, *USENIX Security 2024: 33rd USENIX Security Symposium*, Philadelphia, PA, USA, August 14–16, 2024. USENIX Association.

GMNO18. Rosario Gennaro, Michele Minelli, Anca Nitulescu, and Michele Orrù. Lattice-based zk-SNARKs from square span programs. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 556–573, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

GNS23. Chaya Ganesh, Anca Nitulescu, and Eduardo Soria-Vazquez. Rinocchio: Snarks for ring arithmetic. *J. Cryptol.*, 36(4):41, 2023.

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326, Vienna, Austria, May 8–12, 2016. Springer Berlin Heidelberg, Germany.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019.

IKOS07. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th Annual ACM Symposium on Theory of Computing*, pages 21–30, San Diego, CA, USA, June 11–13, 2007. ACM Press.

KKW18. Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 525–537, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

KLPS26. Erki Külaots, Helger Lipmaa, Roberto Parisella, and Janno Siim. Special soundness and binding properties: A framework for tightly secure zk-SNARKs. In Joan Daemen and Emmanuel Thomé, editors, *Advances in Cryptology – EUROCRYPT 2026, Part VII*, volume 16547 of *Lecture Notes in Computer Science*, pages 239–267, Rome, Italy, May 10–14, 2026. Springer, Cham, Switzerland.

KZ20. Daniel Kales and Greg Zaverucha. Improving the performance of the Picnic signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(4):154–188, 2020.

KZG10. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*,

volume 6477 of *Lecture Notes in Computer Science*, pages 177–194, Singapore, December 5–9, 2010. Springer Berlin Heidelberg, Germany.

LPS25. Helger Lipmaa, Roberto Parisella, and Janno Siim. On knowledge-soundness of plonk in ROM from falsifiable assumptions. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 362–395, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

Nit19. Anca Nitulescu. Lattice-based zero-knowledge snargs for arithmetic circuits. In Peter Schwabe and Nicolas Thériault, editors, *Progress in Cryptology - LATINCRYPT 2019 - 6th International Conference on Cryptology and Information Security in Latin America, Santiago de Chile, Chile, October 2-4, 2019, Proceedings*, volume 11774 of *Lecture Notes in Computer Science*, pages 217–236. Springer, 2019.

Ped92. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, Santa Barbara, CA, USA, August 11–15, 1992. Springer Berlin Heidelberg, Germany.

PNSN26. Nhat-Minh Pham, Khoa Nguyen, Willy Susilo, and Chuong Nguyen. Publicly verifiable secret sharing: Generic constructions and lattice-based instantiations in the standard model. *Computer Standards & Interfaces*, 96:104090, 2026.

RT25. Lior Rotem and Stefano Tessaro. Straight-line knowledge extraction for multi-round protocols. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 95–127, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

YW22. Kang Yang and Xiao Wang. Non-interactive zero-knowledge proofs to multiple verifiers. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part III*, volume 13793 of *Lecture Notes in Computer Science*, pages 517–546, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.

Zac24. Eric A. Zacks. Contract and fraud. *University of Pennsylvania Journal of Business Law*, 26:570, 2024.

Appendix

A Omitted Preliminaries

A.1 Zero-Knowledge Notions

Definition 23 (Honest-Verifier Zero-Knowledge). *An interactive proof system is honest-verifier zero-knowledge if there exists a polynomial-time algorithm* Sim *(simulator) that, for any* $(x, w) \in \mathcal{L}_{RW}$*, distribution of outputs from* $\mathsf{Sim}(x)$ *and that of transcripts observed in* $\langle P(w), V\rangle(x)$ *are indistinguishable.*

B Critical Round in Other Proof Systems

B.1 Cut-and-choose KKW

In this part, we show that the cut-and-choose optimization of KKW can be captured by our CRSS framework, and that the resulting protocol satisfies $(1, 2)$-CRSS with critical round $i^*_{\mathsf{ss}} = 2$. We refer readers to Section C for a detailed description of the KKW framework. In the original work [KKW18], the authors first show that the bare-bones KKW protocol (without parallel repetitions) is $(2, 2)$-special sound, and then extend it to a cut-and-choose version by applying a $(m-\tau)$-out-of-m cut-and-choose technique in the preprocessing phase. We call this version the cut-and-choose KKW (C&C-KKW) protocol. Compared to the bare-bones protocol, C&C-KKW avoids heavy parallel repetitions and has been used to derive short signatures such as Picnic2 [KZ20].

The standard technique to prove soundness with the help of special soundness does not apply to C&C-KKW. Specifically, if one tries to show that C&C-KKW is special sound, it results in $(\binom{m-1}{\tau-1} + 1, 2)$-special soundness, which is rather useless as it includes a parameter exponential in τ. This is because in the online phase, two different challenges in the second round may only have one distinct execution, and in the offline phase, there are at most $\binom{m-1}{\tau-1}$ choices that contain this particular execution. The exponentially large special soundness parameter makes the knowledge extractor inefficient. Furthermore, the same flaw is also observed in the following papers [Beu20, BGKM23] which introduce the notion of Σ-protocols with trusted helper. The trusted helper is eventually removed by a cut-and-choose paradigm, which involves an exponentially large special soundness parameter (e.g. Theorem 3 in [Beu20]).

Our CRSS notion captures C&C-KKW to overcome the above limitation. In particular, we prove that the C&C-KKW protocol satisfies $(2, 2, \boldsymbol{\Phi})$-predicate SS, where the predicate $\boldsymbol{\Phi}$ ensures that, in the extended tree of transcripts, the views and preprocessing materials of at least one MPC instance are fully revealed. The predicate system is parameterized by a threshold $d \in [\tau]$ on the number of MPC instances whose views are fully revealed in the first branch; the parameter trades off the failure densities of the two levels (Lemma 4). We further show that the protocol satisfies $(\vec{k}, \Phi_{cr})$-predicate CRSS with parameters: $i^*_{\mathsf{ss}} = 2$, $\vec{k} = (1, 2)$, and $\Phi_{cr} = \emptyset$. That is, branching from the second round suffices for witness extraction, and no additional predicate is required on the critical subtree, as shown in Figure 1. We note that for bare-bones KKW (without cut-and-choose), both predicates $(\boldsymbol{\Phi}, \Phi_{cr})$ are trivial: bare-bones KKW is $(2, 2)$-special sound and satisfies *plain CRSS* with $i^*_{\mathsf{ss}} = 2$, $\vec{k} = (1, 2)$. This is natural, since without cut-and-choose, no transcript condition is needed for witness extraction.

In the following lemma, we assume the following structure of C&C-KKW protocol (see Appendix C for details). The prover first generates and commits to m executions of the preprocessing stage, and later opens $m - \tau$ of them based on the verifier's challenge. The τ unopened materials are used for executing τ instances of MPC protocol later on. To be clear, for a transcript $(a, \vec{c}_1, z_1, \vec{c}_2, z_2)$, the challenges $\vec{c}_1, \vec{c}_2$ are the set of τ elements. Each hidden preprocessing material is used for one MPC instance, where it opens all-but-one views of parties. Denote $\vec{c}_1 \in [m]^\tau$ as a first round challenge vector, $\vec{c}_1 = (c^1_1, \ldots, c^\tau_1)$, where c^j_1's are pairwise distinct, and $\vec{c}_2 = (c^1_2, \ldots, c^\tau_2) \in [n]^\tau$ as a second round challenge vector, where $c^j_2 \in [n], \forall j \in [\tau]$ (no need to be distinct). In a tree of transcripts, $t_{1,1}$ denotes the left subtree hanging from the root a, including the challenge of first round $\vec{c}_{1,1}$ and two challenges of second round $\vec{c}_{2,1}, \vec{c}_{2,2}$.

For two second-round challenge vectors $\vec{c}, \vec{c}' \in [n]^\tau$, let

$$D(\vec{c}, \vec{c}') := \{j \in [\tau] \mid c^j \neq c'^j\}$$

denote the set of MPC executions on which they differ, and let $d_{\mathsf{H}}(\vec{c}, \vec{c}') := |D(\vec{c}, \vec{c}')|$ be their Hamming distance. In a $(1,2)$-subtree $t_{1,1}$ with second-round challenges $\vec{c}_{2,1}, \vec{c}_{2,2}$, the executions in $D(\vec{c}_{2,1}, \vec{c}_{2,2})$ are exactly those whose n party views are all opened, and we write

$$I(t_{1,1}) := \{c^j_{1,1} \mid j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})\} \subseteq [m]$$

for the indices of the corresponding (unopened) preprocessing executions. Below, a first-round challenge vector $\vec{c}_{1,2}$ is identified with the set of its entries.

Lemma 3. *Let Π be the cut-and-choose KKW protocol described in Appendix C, and let $d \in [\tau]$. Assuming that the hash function used is collision-resistant and that the commitment scheme used is computationally binding, then the proof system Π is:*

- *Extended predicate $\vec{k}$-special sound where $\vec{k} = (2,2)$, and $\boldsymbol{\Phi} = \boldsymbol{\Phi}^{(d)}$ is defined as follows:*
  - *$\Phi^{\mathsf{chal}}_{2,2}(t_{2,1}, \vec{c}_{2,2}) = 1$ if and only if $d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2}) \geq d$, where $\vec{c}_{2,1}$ is the second-round challenge of the transcript $t_{2,1}$. That is, the two second-round challenges in a $(1,2)$-subtree must differ on at least d MPC executions, so that the views of at least d executions are fully opened.*
  - *$\Phi^{\mathsf{chal}}_{1,2}(t_{1,1}, \vec{c}_{1,2}) = 1$ if and only if $I(t_{1,1}) \not\subseteq \vec{c}_{1,2}$. That is, the first-round challenge of the second branch must leave* opened *the preprocessing of at least one execution whose n views are opened in $t_{1,1}$.*
  - *All other predicates ($\Phi^{\mathsf{chal}}_{1,1}$ and $\Phi^{\mathsf{chal}}_{2,1}$) always return 1.*
- *$\vec{k}$-critical-round special sound where $i^*_{\mathsf{ss}} = 2$, $\vec{k} = (1,2)$, with Φ_{cr} trivial (always returns 1).*

Proof. **Extended Predicate SS.** We briefly explain the predicate $\boldsymbol{\Phi}$ defined in the construction. The predicate is used to guarantee the witness extraction is possible. Recall that the prover creates m preprocessing states and opens $m - \tau$ of them in the first round. In the second round the prover runs τ corresponding MPC instances each with n parties. Observe that as long as there exists an instance such that the prover opens both the preprocessing state and all corresponding MPC parties, the extraction will succeed. The level-2 predicate $\Phi^{\mathsf{chal}}_{2,2}(t_{2,1}, \vec{c}_{2,2})$ requires that the two second-round challenges $\vec{c}_{2,1}, \vec{c}_{2,2}$ of the $(1,2)$-subtree $t_{1,1}$ differ on at least $d \geq 1$ executions, i.e., $d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2}) \geq d$; for each $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$ all n party views of the j-th MPC execution are opened in $t_{1,1}$, and $I(t_{1,1}) = \{c^j_{1,1} \mid j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})\}$ collects the indices of the corresponding preprocessing executions. The level-1 predicate $\Phi^{\mathsf{chal}}_{1,2}(t_{1,1}, \vec{c}_{1,2})$ then checks whether the challenge vector $\vec{c}_{1,2}$ covers the set $I(t_{1,1})$. Since the indices in $\vec{c}_{1,2}$ represent unopened states, covering $I(t_{1,1})$ means that the extraction would fail, and $\Phi^{\mathsf{chal}}_{1,2}$ returns 0 in this case.

Next we construct the extended extractor ExExt given a $(2,2)$-tree T and $\boldsymbol{\Phi}(T) = 1$. We refer the reader to Appendix C for a detailed description of the construction of C&C KKW having τ MPC executions. However, we note that extracting a valid witness only requires the views and preprocessing material corresponding to a single special MPC execution.

Extended Extractor. We construct the extended extractor ExExt given a $(2,2)$-tree T and $\boldsymbol{\Phi}(T) = 1$. The subtree $t_{1,1}$ includes two accepting transcripts $(a, \vec{c}_{1,1}, z_{1,1}, \vec{c}_{2,1}, z_{2,1})$ and $(a, \vec{c}_{1,1}, z_{1,1}, \vec{c}_{2,2}, z_{2,2})$, ExExt does the following:

- Given another transcript $(a, \vec{c}_{1,2}, z_{1,2}, *, *)$, ExExt first checks if the predicate $\boldsymbol{\Phi}^{(d)}$ is satisfied or not. If satisfied, then $|I(t_{1,1})| \geq d \geq 1$ by $\Phi^{\mathsf{chal}}_{2,2}$ and $I(t_{1,1}) \not\subseteq \vec{c}_{1,2}$ by $\Phi^{\mathsf{chal}}_{1,2}$, so ExExt can pick an index $j \in I(t_{1,1})$ with $j \notin \vec{c}_{1,2}$. Since $j \notin \vec{c}_{1,2}$, the preprocessing material of j-th MPC execution is opened, and $j \in I$ leads to $c^j_{2,1} \neq c^j_{2,2}$ then for j-th MPC execution the views of MPC parties are opened. It means that in j-th MPC execution, for both preprocessing phase and online phase are all opened.
- As $j \notin \vec{c}_{1,2}$, the extractor ExExt can recover seed_j from $z_{1,2}$, from seed_j, ExExt computes $\{\mathsf{state}_{j,i}\}_{i\in[n]}$ that are used to mask witness w. Therefore from the masked value $(\hat{z}^j_\alpha)_{i\in[n]}$ and $\{\mathsf{state}_{j,i}\}_{i\in[n]}$, ExExt can extract a witness w.

– The extracted witness is an actual witness. Consider the subtree $t_{1,1}$, $\{\mathsf{state}_{j,i}\}_{i\in[n]}$ computed by ExExt must be consistent with the values outputted in the fifth move, that are responses for two challenges $c^j_{2,1}$ and $c^j_{2,2}$. Otherwise, if this consistency fails, it would imply either a break in the binding property of the commitment scheme or a collision in the hash function H. In such a case, as a solution to R_{hd}, the extractor would output either a hash collision or the inputs required to break the commitment scheme, since all states are committed using a fixed value $a := h$.

Critical Round SS. We now show that C&C KKW satisfies critical round special soundness. Given a $(1,2)$-tree of transcripts T^*, our goal is to show that there exists a critical extractor CrExt that either extracts an actual witness from T^*, or otherwise every extended $(2,2)$-tree $T \sqsupseteq T^*$ with $\boldsymbol{\Phi}^{(d)}(T) = 1$ yields a solution to the hardness assumption via ExExt, as required by Definition 14.

Critical-round Extractor. Given two accepting transcripts $(a, \vec{c}_{1,1}, z_{1,1}, \vec{c}_{2,1}, z_{2,1})$ and $(a, \vec{c}_{1,1}, z_{1,1}, \vec{c}_{2,2}, z_{2,2})$, the extractor CrExt works as follows:

– Since $\vec{c}_{2,1} \neq \vec{c}_{2,2}$, the set $D(\vec{c}_{2,1}, \vec{c}_{2,2})$ of indices j with distinct challenges $c^j_{2,1} \neq c^j_{2,2}$ is non-empty. CrExt performs the following steps for every $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$ (at most τ of them) and outputs the first candidate w satisfying $R(x, w) = 1$, if any. We now pay attention to the j-th execution of MPC.
– From $z_{1,1}$, CrExt learns $\{\hat{z}^j_\alpha\}$ that is the masked value of witness in j-th execution of MPC.
– Since $c^j_{2,1} \neq c^j_{2,2}$ then from $z_{2,1}$ and $z_{2,2}$ together, CrExt learns all $\{\mathsf{state}_{j,i}\}_{i\in[n]}$.
– From $\{\mathsf{state}_{j,i}\}_{i\in[n]}$, CrExt learns the value λ^j_α as the masks of each input wire, therefore from $\{\hat{z}^j_\alpha\}$, CrExt can effectively compute a candidate witness w_j.

Fall-back to the extended extractor. It remains to verify the "otherwise" clause of Definition 14 against the predicate system $\boldsymbol{\Phi}^{(d)}$. Recall that $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2}) \subseteq [\tau]$ indexes an online MPC execution, while $c^j_{1,1} \in [m]$ is the index of the preprocessing execution it uses; since the entries of $\vec{c}_{1,1}$ are pairwise distinct, $j \mapsto c^j_{1,1}$ is a bijection from $D(\vec{c}_{2,1}, \vec{c}_{2,2})$ onto $I(T^*)$. Suppose CrExt fails, i.e., $R(x, w_j) = 0$ for every $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$, and let $T \sqsupseteq T^*$ be any extended $(2,2)$-tree with $\boldsymbol{\Phi}^{(d)}(T) = 1$, whose second branch has first-round challenge $\vec{c}_{1,2}$. By $\Phi^{\mathsf{chal}}_{1,2}$ there is $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$ with $c^j_{1,1} \notin \vec{c}_{1,2}$, so $\mathsf{seed}_{c^j_{1,1}}$ is opened in $z_{1,2}$ and ExExt recomputes $\{\mathsf{state}^*_{c^j_{1,1},i}\}_{i\in[n]}$ from it. The two transcripts of T^* together open all n states $\{\mathsf{state}_{c^j_{1,1},i}\}_{i\in[n]}$ and all n broadcast messages $\{\mathsf{msgs}_i^{c^j_{1,1}}\}_{i\in[n]}$ of execution j, each party's messages being opened in at least one of them. If the broadcast messages opened for the same party in the two transcripts differ, they are two distinct preimages of the digest h' fixed in $z_{1,1}$, and ExExt outputs a collision of H. If the recomputed states $\{\mathsf{state}^*_{c^j_{1,1},i}\}_{i\in[n]}$ differ from the opened states $\{\mathsf{state}_{c^j_{1,1},i}\}_{i\in[n]}$, then both are consistent with the commitments $\{com_{c^j_{1,1},i}\}_{i\in[n]}$ and the hashes $h_{c^j_{1,1}}$ and h verified in the accepting transcripts, and ExExt outputs either a collision of H or a break of the binding of Com. Otherwise, execution j is an honest MPC evaluation of C under correctly generated preprocessing on the masked input $\{\hat{z}^j_\alpha\}$ with a single consistent set of broadcast messages, so the verifier's reconstruction in either transcript recomputes the honest execution; since both transcripts are accepting, the output bit is 1, hence $C(w_j) = 1$, contradicting $R(x, w_j) = 0$. In every case ExExt outputs a solution s with $R_{hd}(y, s) = 1$.

□

Next, we compute the failure densities of the above predicate system. Recall from Definition 11 that a failure density is a worst-case bound over *every* good tuple of subtrees, and that a good $(1,2)$-subtree $t_{1,1} \in \mathsf{GoodTree}_{1,1}$ must itself satisfy $\boldsymbol{\Phi}^{(d)}_2(t_{1,1}) = 1$, i.e., $\Phi^{\mathsf{chal}}_{2,2}$ holds inside it.

Lemma 4. *For every $d \in [\tau]$, the predicate system $\boldsymbol{\Phi}^{(d)}$ of Lemma 3 has the following failure densities:*

$$p^{\mathsf{chal}}_{1,1} = p^{\mathsf{chal}}_{2,1} = 0, \qquad p^{\mathsf{chal}}_{2,2} = \frac{\sum_{i=1}^{d-1} \binom{\tau}{i}(n-1)^i}{n^\tau}, \qquad p^{\mathsf{chal}}_{1,2} = \frac{\binom{m-d}{\tau-d}}{\binom{m}{\tau}} = \frac{\binom{m-d}{m-\tau}}{\binom{m}{m-\tau}},$$

where the empty sum for $d = 1$ equals 0.

Proof. The predicates $\Phi^{\mathsf{chal}}_{1,1}$ and $\Phi^{\mathsf{chal}}_{2,1}$ are trivial and have no bad challenges, so $p^{\mathsf{chal}}_{1,1} = p^{\mathsf{chal}}_{2,1} = 0$.

Level 2 predicate. Fix any accepting transcript $t_{2,1} \in \mathsf{GoodTree}_{2,1}$ with second-round challenge $\vec{c}_{2,1} \in [n]^\tau$. A challenge $\vec{c} \in \mathcal{C}_2 \setminus \{\vec{c}_{2,1}\}$ is bad for $\Phi^{\mathsf{chal}}_{2,2}$ if and only if $1 \le d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}) \le d-1$. The number of vectors at Hamming distance exactly i from $\vec{c}_{2,1}$ is $\binom{\tau}{i}(n-1)^i$: choose the i coordinates on which they differ and, on each, one of the $n-1$ other party indices. Hence $|\mathsf{BadChal}_{2,2}(t_{2,1})| = \sum_{i=1}^{d-1}\binom{\tau}{i}(n-1)^i$, independently of $t_{2,1}$, and dividing by $|\mathcal{C}_2| = n^\tau$ gives $p^{\mathsf{chal}}_{2,2}$. For $d=1$ every challenge distinct from $\vec{c}_{2,1}$ is good, and $p^{\mathsf{chal}}_{2,2} = 0$.

Level 1 predicate. Fix any $t_{1,1} \in \mathsf{GoodTree}_{1,1}$. By Definition 11, $t_{1,1}$ satisfies $\boldsymbol{\Phi}^{(d)}_2(t_{1,1}) = 1$, so its second-round challenges satisfy $\Phi^{\mathsf{chal}}_{2,2}(t_{2,1}, \vec{c}_{2,2}) = 1$, i.e., $d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2}) \ge d$. Since the entries of $\vec{c}_{1,1}$ are pairwise distinct, $i := |I(t_{1,1})| = d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2}) \ge d$. A first-round challenge $\vec{c} \in \mathcal{C}_1 \setminus \{\vec{c}_{1,1}\}$ is bad for $\Phi^{\mathsf{chal}}_{1,2}$ if and only if $I(t_{1,1}) \subseteq \vec{c}$. Viewing first-round challenges as τ-subsets of $[m]$ (the predicates depend on $\vec{c}$ only through its underlying set, so the ordered representation of Appendix C scales both $|\mathsf{BadChal}_{1,2}|$ and $|\mathcal{C}_1|$ by the same factor $\tau!$ and leaves the density unchanged), the τ-subsets containing a fixed i-subset number $\binom{m-i}{\tau-i}$; one of them is $\vec{c}_{1,1}$ itself, which Definition 11 excludes. Thus

$$|\mathsf{BadChal}_{1,2}(t_{1,1})| = \binom{m-i}{\tau-i} - 1 \le \binom{m-i}{\tau-i}.$$

This count is non-increasing in i, since $\binom{m-i-1}{\tau-i-1}\big/\binom{m-i}{\tau-i} = \frac{\tau-i}{m-i} \le 1$, so the worst case over all good $t_{1,1}$ is $i=d$, and $|\mathsf{BadChal}_{1,2}(t_{1,1})| \le \binom{m-d}{\tau-d}$ for every $t_{1,1} \in \mathsf{GoodTree}_{1,1}$. Dividing by $|\mathcal{C}_1| = \binom{m}{\tau}$ yields $p^{\mathsf{chal}}_{1,2}$; the second expression in the lemma follows from $\binom{m-d}{\tau-d} = \binom{m-d}{m-\tau}$ and $\binom{m}{\tau} = \binom{m}{m-\tau}$.

We stress that both bounds hold for *every* good subtree, as Definition 11 demands. Note that the role of the level-2 predicate $\Phi^{\mathsf{chal}}_{2,2}$ is to exclude the subtrees with $|I(t_{1,1})| < d$, on which $\Phi^{\mathsf{chal}}_{1,2}$ would have a larger bad set. □

Corollary 4. *Let Π be the 5-move C&C-KKW protocol of Appendix C with $(m-\tau)$-out-of-m cut-and-choose in the preprocessing phase and n MPC parties, and let $d \in [\tau]$. Assuming that the hash function is collision-resistant and the commitment scheme is computationally binding, the Fiat-Shamir transform $\mathrm{FS}[\Pi]$ is knowledge sound for the extended relation with knowledge error*

$$\kappa_{\mathsf{c\&c}}(d) = 2(Q+1)\left[\frac{\binom{m-d}{\tau-d}}{\binom{m}{\tau}} + \frac{\max\left(1,\ \sum_{i=1}^{d-1}\binom{\tau}{i}(n-1)^i\right)}{n^\tau}\right],$$

where Q is the number of random oracle queries made by the prover; we write $\kappa_{\mathsf{c\&c}} := \min_{d\in[\tau]}\kappa_{\mathsf{c\&c}}(d)$. The knowledge extractor invokes the prover at most $4+3Q$ times in expectation. In particular, if $m \ge 2\tau$ then

$$\kappa_{\mathsf{c\&c}} \le \kappa_{\mathsf{c\&c}}(\lceil \tau/4 \rceil) \le 2(Q+1)\left[2^{-\tau/4} + e^{-\tau/16}\right],$$

which is negligible in τ.

Proof. By Lemma 3, Π is $((2,2), \boldsymbol{\Phi}^{(d)})$-predicate special sound for the extended relation, and by Lemma 4 the failure densities are $p^{\mathsf{chal}}_{1,1} = p^{\mathsf{chal}}_{2,1} = 0$, $p^{\mathsf{chal}}_{1,2} = \binom{m-d}{\tau-d}/\binom{m}{\tau}$ and $p^{\mathsf{chal}}_{2,2} = \sum_{i=1}^{d-1}\binom{\tau}{i}(n-1)^i/n^\tau$. Both chains of predicates are efficiently checkable, since $\Phi^{\mathsf{chal}}_{2,2}$ and $\Phi^{\mathsf{chal}}_{1,2}$ depend on their inputs only through the challenge vectors, which they compare coordinate-wise. Hence Theorem 1 applies with $k_1 = k_2 = 2$, $|\mathcal{C}_1| = \binom{m}{\tau}$ (counting first-round challenges as τ-subsets, as in Lemma 4) and $|\mathcal{C}_2| = n^\tau$, and yields the knowledge error

$$2(Q+1)\left[\max\left(\frac{1}{\binom{m}{\tau}},\ p^{\mathsf{chal}}_{1,2}\right) + \max\left(\frac{1}{n^\tau},\ p^{\mathsf{chal}}_{2,2}\right)\right] = \kappa_{\mathsf{c\&c}}(d),$$

where the first maximum equals $p^{\mathsf{chal}}_{1,2}$ because $\binom{m-d}{\tau-d} \ge 1$. The expected number of prover invocations is $K + Q(K-1) = 4+3Q$ for $K = k_1k_2 = 4$. A solution to the hard problem output by the extended extractor is a hash collision or a binding break, so the extractor outputs a witness for R except with the (negligible) advantage against collision resistance or binding.

It remains to bound $\kappa_{\mathsf{c\&c}}(d)$ for $d = \lceil \tau/4 \rceil$. The level-1 term satisfies $\binom{m-d}{\tau-d}/\binom{m}{\tau} = \prod_{i=0}^{d-1}\frac{\tau-i}{m-i} \le (\tau/m)^d \le 2^{-\tau/4}$ for $m \ge 2\tau$. For the level-2 term, observe that $\max(1, \sum_{i=1}^{d-1}\binom{\tau}{i}(n-1)^i)/n^\tau \le$

$\sum_{i=0}^{d-1} \binom{\tau}{i}(n-1)^i/n^\tau$, and that the right-hand side is the probability that a uniformly random $\vec{c} \in [n]^\tau$ lies within Hamming distance $d-1$ of a fixed vector. Each coordinate of $\vec{c}$ differs from the fixed vector independently with probability $1-\frac{1}{n}$, so this is $\Pr[X \le d-1]$ for $X \sim \mathrm{Bin}(\tau, 1-\frac{1}{n})$ with mean $\mu = \tau(1-\frac{1}{n}) \ge \tau/2$. Since $d-1 < \tau/4 \le \mu/2$, the Chernoff bound $\Pr[X \le \mu/2] \le e^{-\mu/8}$ gives $\Pr[X \le d-1] \le e^{-\tau/16}$. □

Remark 3 (The additive bound is non-tight). The knowledge error $\kappa_{\mathsf{c\&c}} = \min_d \kappa_{\mathsf{c\&c}}(d)$ of Corollary 4 is negligible in τ (for $m \ge 2\tau$) but it is far from tight. The failure densities of the two levels appear in Theorem 1 *additively*, whereas the soundness error of C&C-KKW given in [KKW18] (see Section C.2), $\kappa^* = \max_{0 \le i \le \tau} \binom{m-i}{m-\tau} / \binom{m}{m-\tau} \cdot n^{-(\tau-i)}$, is a *product* of an offline and an online term. Consequently $\kappa_{\mathsf{c\&c}}$ is roughly twice as lossy in bits. Ignoring the common factor $2(Q+1)$, for the Picnic2 parameter sets (m, n, τ) of [KZ20] we obtain:

(m, n, τ)	$-\log_2 \kappa^*$	$-\log_2 \kappa_{\mathsf{c\&c}}/(2(Q+1))$	optimal d
$(343, 64, 27)$ (L1)	128.06	56.76	14
$(570, 64, 39)$ (L3)	192.06	83.86	20
$(803, 64, 50)$ (L5)	252.01	108.93	25

Equivalently, for $n = 64$ and $m = 631$, reaching 2^{-128} through $\kappa_{\mathsf{c\&c}}$ requires $\tau = 66$ online executions instead of $\tau = 23$ through κ^*. The loss is inherent to bounding the two levels separately; recovering κ^* requires an extractor that resamples the two rounds jointly, outside the PSS framework, which we leave for future work. We stress that the tightness of the failure-density bound is orthogonal to the benefit of CRSS. The Fiat-Shamir extractor of Corollary 4 still builds a full $(2,2)$-tree, but the threshold witness sharing construction of Section 4.3 uses the $(1,2)$-critical tree, whose size and predicate-freeness are unaffected by the loss above.

B.2 Σ-Protocols with Helper

The analysis in Section B.1 is not specific to KKW: it uses only the cut-and-choose skeleton, and never the MPC-in-the-head instantiation of the 3-move protocol that is repeated inside it. We illustrate this on Σ-protocols with helper [Beu20].

A *Σ-protocol with helper* for a relation R is a 3-move public-coin protocol augmented with a trusted setup: on a uniform seed seed, the helper computes $aux \leftarrow \mathsf{Setup}(\mathsf{seed})$ and hands aux to the verifier and seed to the prover. Its 2-special soundness is *conditional* on the setup being honest: given $aux = \mathsf{Setup}(\mathsf{seed})$ for some seed seed and two accepting transcripts (α, γ, ζ), $(\alpha, \gamma', \zeta')$ with $\gamma \ne \gamma'$ in the challenge space Ch, an extractor outputs a witness for x. Nothing is guaranteed when aux is outside the image of Setup, which is precisely why the helper cannot simply be dropped.

The helper is removed by cut-and-choose, batched exactly as in C&C-KKW [Beu20]. Write Π for the resulting 5-move protocol, with parameters m (setups), τ (unopened setups) and the underlying Σ-protocol Σ:

1. P samples $\mathsf{seed}_1, \dots, \mathsf{seed}_m$, sets $aux_i \leftarrow \mathsf{Setup}(\mathsf{seed}_i)$, and sends $a := H(com_1, \dots, com_m)$, where com_i commits to (seed_i, aux_i).
2. V sends $\vec{c}_1 = (c_1^1, \dots, c_1^\tau) \in [m]^\tau$ with pairwise distinct entries: the setups to be kept *unopened*.
3. P sends z_1, consisting of $\{\mathsf{seed}_i\}_{i \notin \vec{c}_1}$, the openings of $\{aux_{c_1^j}\}_{j \in [\tau]}$, and the first messages $\alpha^1, \dots, \alpha^\tau$ of τ executions of Σ, the j-th run under $aux_{c_1^j}$.
4. V sends $\vec{c}_2 = (c_2^1, \dots, c_2^\tau) \in \mathsf{Ch}^\tau$, each coordinate drawn *independently* and uniformly from Ch.
5. P sends $z_2 = (\zeta^1, \dots, \zeta^\tau)$, the response of the j-th execution to c_2^j.

V accepts if every opened seed is consistent with a (recomputing $\mathsf{Setup}(\mathsf{seed}_i)$ and com_i) and every $(\alpha^j, c_2^j, \zeta^j)$ is accepting under $aux_{c_1^j}$. This is the structure of C&C-KKW with one MPC execution replaced by one Σ-execution, so $|\mathcal{C}_1| = \binom{m}{\tau}$ and $|\mathcal{C}_2| = |\mathsf{Ch}|^\tau$, the role of the number n of MPC parties being played by $|\mathsf{Ch}|$. As for C&C-KKW, plain special soundness of Π is useless: two second-round challenges may differ in a single execution j, and $\binom{m-1}{\tau-1}$ first-round challenges keep the setup of that execution unopened, so the naive parameter is $(\binom{m-1}{\tau-1} + 1, 2)$, exponential in τ.

We reuse $D(\cdot,\cdot)$, d_{H} and $I(\cdot)$ of the previous subsection verbatim, read in the present setting: for a $(1,2)$-subtree $t_{1,1}$ with second-round challenges $\vec{c}_{2,1}, \vec{c}_{2,2}$, the set $D(\vec{c}_{2,1}, \vec{c}_{2,2}) = \{j \in [\tau] \mid c_{2,1}^j \ne c_{2,2}^j\}$

collects the executions on which the two branches answer *distinct* Σ-challenges—that is, those to which 2-special soundness applies, playing the role of the executions whose n party views are all opened in KKW—and $I(t_{1,1}) = \{c^j_{1,1} \mid j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})\} \subseteq [m]$ collects the indices of the setups they use.

Corollary 5. *Let Σ be a 2-special sound Σ-protocol with helper for R with challenge space Ch, $|\mathsf{Ch}| \geq 2$, let Π be its $(m-\tau)$-out-of-m batched cut-and-choose as above, and let $d \in [\tau]$. Assume that the hash function H is collision resistant and that the commitment scheme Com is computationally binding, and let R_{hd} be the relation of collisions of H and binding breaks of Com. Then Π is*

- *extended $((2,2), \boldsymbol{\Phi}^{(d)})$-predicate special sound, for the predicate system $\boldsymbol{\Phi}^{(d)}$ of Lemma 3 read as above: $\Phi^{\mathsf{chal}}_{2,2}(t_{2,1}, \vec{c}_{2,2}) = 1$ if and only if $d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2}) \geq d$, $\Phi^{\mathsf{chal}}_{1,2}(t_{1,1}, \vec{c}_{1,2}) = 1$ if and only if $I(t_{1,1}) \not\subseteq \vec{c}_{1,2}$, and $\Phi^{\mathsf{chal}}_{1,1}, \Phi^{\mathsf{chal}}_{2,1}$ are trivial; and*
- *$(\vec{k}, \Phi_{cr})$-critical-round special sound at round $i^*_{\mathsf{ss}} = 2$ with $\vec{k} = (1,2)$ and Φ_{cr} trivial.*

Its failure densities are those of Lemma 4 with n replaced by $|\mathsf{Ch}|$.

Proof (Sketch.). Write the two transcripts of the $(1,2)$-critical tree T^* as $(a, \vec{c}_{1,1}, z_{1,1}, \vec{c}_{2,1}, z_{2,1})$ and $(a, \vec{c}_{1,1}, z_{1,1}, \vec{c}_{2,2}, z_{2,2})$, and let $t_{1,1} := T^*$.

Critical-round extractor. The two transcripts share $z_{1,1}$, hence for every $j \in [\tau]$ the same setup $aux_{c^j_{1,1}}$ and the same Σ first message α^j. For $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$ the responses ζ^j in $z_{2,1}$ and $z_{2,2}$ are therefore two accepting transcripts of Σ on a common first message with distinct challenges, under the common $aux_{c^j_{1,1}}$. CrExt runs the 2-special soundness extractor of Σ for *every* such j (at most τ of them) and outputs the first candidate w_j with $R(x, w_j) = 1$, if any. Since $\vec{c}_{2,1} \neq \vec{c}_{2,2}$ we have $D(\vec{c}_{2,1}, \vec{c}_{2,2}) \neq \emptyset$, so Φ_{cr} may be taken trivial.

Fall-back to the extended extractor. Suppose $R(x, w_j) = 0$ for every $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$, and let $T \sqsupseteq T^*$ be any $(2,2)$-tree with $\boldsymbol{\Phi}^{(d)}(T) = 1$, whose second branch has first-round challenge $\vec{c}_{1,2}$ and response $z_{1,2}$. Since the entries of $\vec{c}_{1,1}$ are pairwise distinct, $j \mapsto c^j_{1,1}$ is a bijection from $D(\vec{c}_{2,1}, \vec{c}_{2,2})$ onto $I(t_{1,1})$; by $\Phi^{\mathsf{chal}}_{2,2}$ we have $|I(t_{1,1})| = d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2}) \geq d \geq 1$, and by $\Phi^{\mathsf{chal}}_{1,2}$ there is $j \in D(\vec{c}_{2,1}, \vec{c}_{2,2})$ with $c^j_{1,1} \notin \vec{c}_{1,2}$. The setup $c^j_{1,1}$ is thus opened in the second branch, so ExExt reads $\mathsf{seed}_{c^j_{1,1}}$ off $z_{1,2}$ and recomputes $aux^* := \mathsf{Setup}(\mathsf{seed}_{c^j_{1,1}})$, while $t_{1,1}$ reveals $aux_{c^j_{1,1}}$; both are verified against the same commitment $com_{c^j_{1,1}}$ fixed by the digest a. If $aux^* \neq aux_{c^j_{1,1}}$, these are two distinct openings of $com_{c^j_{1,1}}$, and ExExt outputs a break of the binding of Com or a collision of H. Otherwise $aux_{c^j_{1,1}}$ is honestly generated, so the 2-special soundness of Σ applies to the two transcripts above and w_j is a witness for x, contradicting $R(x, w_j) = 0$. In either case ExExt outputs s with $R_{hd}(y, s) = 1$, as Definition 14 requires. Running the same argument on the leftmost $(1,2)$-subtree of an arbitrary $(2,2)$-tree T with $\boldsymbol{\Phi}^{(d)}(T) = 1$ gives ExExt itself, hence extended predicate special soundness.

Failure densities. The proof of Lemma 4 uses only combinatorics. At level 1 it counts τ-subsets of $[m]$ containing a fixed i-subset, which is unchanged, and its one structural input—that $|I(t_{1,1})| = d_{\mathsf{H}}(\vec{c}_{2,1}, \vec{c}_{2,2})$—uses only the pairwise distinctness of the entries of $\vec{c}_{1,1}$, which holds here. At level 2 it counts the vectors at Hamming distance i from a fixed $\vec{c}_{2,1}$, which is $\binom{\tau}{i}(|\mathsf{Ch}| - 1)^i$ out of $|\mathsf{Ch}|^\tau$ instead of $\binom{\tau}{i}(n-1)^i$ out of n^τ, since each coordinate ranges over Ch independently. □

Two consequences follow, and it is worth keeping them apart, as in Remark 3. First, Fiat-Shamir knowledge extraction for Π needs a $(2,2)$-tree rather than the $(\binom{m-1}{\tau-1} + 1, 2)$-tree that plain special soundness demands, which is what makes the extractor run in polynomial time. Second, its critical-round extractor needs only the $(1,2)$-critical tree, whose size and predicate-freeness are independent of m and τ.

Corollary 6. *In the setting of Corollary 5, the Fiat-Shamir transform $\mathrm{FS}[\Pi]$ is knowledge sound for the extended relation with knowledge error $\kappa_{\mathsf{helper}} := \min_{d \in [\tau]} \kappa_{\mathsf{helper}}(d)$, where*

$$\kappa_{\mathsf{helper}}(d) = 2(Q+1)\left[\frac{\binom{m-d}{\tau-d}}{\binom{m}{\tau}} + \frac{\max\left(1,\ \sum_{i=1}^{d-1}\binom{\tau}{i}(|\mathsf{Ch}|-1)^i\right)}{|\mathsf{Ch}|^\tau}\right]$$

and Q is the number of random oracle queries made by the prover. The knowledge extractor invokes the prover at most $4+3Q$ times in expectation. In particular, if $m \geq 2\tau$ then $\kappa_{\mathsf{helper}} \leq 2(Q+1)\left[2^{-\tau/4} + e^{-\tau/16}\right]$, which is negligible in τ.

Proof. Identical to the proof of Corollary 4. Plug the failure densities of Corollary 5 into Theorem 1 with $k_1 = k_2 = 2$, $|\mathcal{C}_1| = \binom{m}{\tau}$ and $|\mathcal{C}_2| = |\mathsf{Ch}|^\tau$; both chains of predicates are efficiently checkable, since $\Phi^{\mathsf{chal}}_{2,2}$ and $\Phi^{\mathsf{chal}}_{1,2}$ depend on their inputs only through the challenge vectors, which they compare coordinate-wise. The first maximum of Theorem 1 equals $p^{\mathsf{chal}}_{1,2}$ because $\binom{m-d}{\tau-d} \geq 1$. The asymptotic bound is the computation of Corollary 4 at $d = \lceil \tau/4 \rceil$, which uses only $|\mathsf{Ch}| \geq 2$. □

Remark 4. As in Remark 3, the two levels enter Theorem 1 additively, so κ_{helper} is lossy in the same way and by the same factor: reaching a target error costs roughly twice the bits of security that a product-form bound would.

This covers the MQ-based and PKP-based schemes of [Beu20] and the code-based signatures of [BGKM23], whose underlying proofs of knowledge follow the helper paradigm.

B.3 KZG-Compiled Polynomial IOP

Külaots et al. [KLPS26] define special soundness for polynomial IOPs and prove that compiling a special-sound polynomial IOP with a suitably binding polynomial commitment scheme yields a computationally special-sound interactive argument. For the KZG scheme [KZG10], these notions are falsifiable. We show that their running example, a batched Boolean check compiled with KZG, satisfies plain CRSS (Definition 14). The commitments are deterministic and binding, so the statement is fixed before the interaction and the complete witness is determined by the query round alone.

The compiled Boolean check. Let $H \subset \mathbb{F}^*$ be a multiplicative subgroup of order n with generator ω. Let Z_H be its vanishing polynomial, and let $V(Y) := Y(Y-1)$. Let $\mathsf{Com}(\mathsf{ck}, \cdot)$ denote the KZG commitment over a bilinear group for polynomials of degree at most n. The commitment is a deterministic function of the polynomial, and evaluation claims are verified by a pairing check on an evaluation proof. We use the standard single-group-element proof for each opening, canonical field and group encodings, and group-membership checks. Let $p = |\mathbb{F}|$ be the prime group order, and assume $n \geq 1$, $2n+1 \leq p$, and polynomially bounded n. The hardness relation $R_{hd} = R_{\mathsf{bind}}(\mathsf{ck}, \cdot)$ collects violations of two binding notions of [KLPS26]. A witness consists of a commitment C and verifying openings at more than n pairwise distinct points. Interpolate their claimed values to a polynomial f. A *degree-binding* violation has $\deg f > n$. An *interpolation-binding* violation has $\deg f \leq n$ and $\mathsf{Com}(\mathsf{ck}, f) \neq C$. The witness includes the corresponding tag and all evaluation proofs, so it is efficiently checkable. Degree binding also implies plain polynomial binding by the direct reduction in Lemma 5 below. Both notions are falsifiable and hold for KZG under ModDHE-type assumptions [KLPS26].

The argument is for the commit-and-prove relation

$$R_{\mathsf{bool}} = \left\{ \left((\mathsf{ck}, C_f, C_g),\ (\hat{f}, \hat{g})\right) \ :\ \begin{array}{l} C_f = \mathsf{Com}(\mathsf{ck}, \hat{f}) \ \wedge\ C_g = \mathsf{Com}(\mathsf{ck}, \hat{g}) \\ \wedge\ \hat{f}(\omega^k), \hat{g}(\omega^k) \in \{0,1\} \text{ for all } k \in [n] \end{array} \right\},$$

where $\hat{f}, \hat{g} \in \mathbb{F}[X]$ have degree at most n. The statement is fixed before the interaction and the witness contains no randomness. Since Com is deterministic, R_{bool} is efficiently checkable. On statement (ck, C_f, C_g), the argument proceeds as follows. The initial prover message is empty, so the argument is a five-move protocol with two challenge rounds.

1. (Verifier) Send $\alpha \leftarrow\!\!\$\ \mathbb{F}$.
2. (Prover) Compute the quotient $q := (V(\hat{f}) + \alpha V(\hat{g}))/Z_H$, which has degree at most n, and send $C_q = \mathsf{Com}(\mathsf{ck}, q)$.
3. (Verifier) Send $z \leftarrow\!\!\$\ \mathbb{F}$.
4. (Prover) Send the evaluations $\bar{f} = \hat{f}(z)$, $\bar{g} = \hat{g}(z)$, $\bar{q} = q(z)$ together with their evaluation proofs.

The verifier accepts iff the three evaluation proofs verify and $V(\bar{f}) + \alpha V(\bar{g}) = \bar{q} \cdot Z_H(z)$.

All challenge predicates are trivial, so we are in the plain CRSS setting of Definition 14 and drop the predicate systems from the notation. The critical tree shape is $(1, 2n+1)$, branching only at the query round, and the critical round is $i^*_{\mathsf{ss}} = 2$. The extended tree has $2(2n+1)$ leaves. The $2n+1$ query branches establish the polynomial identity of degree at most $2n$ in the extended extraction.

Lemma 5 (Plain binding from degree binding). *Given distinct degree-at-most-n polynomials f, g with $\mathsf{Com}(\mathsf{ck}, f) = \mathsf{Com}(\mathsf{ck}, g)$, one can deterministically produce a degree-binding violation in polynomial time, without a success-probability loss.*

Proof. Among any $n+1$ fixed distinct field points, find z_* with $f(z_*) \neq g(z_*)$; such a point exists because $f - g$ is a nonzero polynomial of degree at most n. Choose $n+1$ further distinct points, none equal to z_*. There are enough points since $n + 2 \leq 2n + 1 \leq p$. Use the public KZG opening algorithm to open f at those $n+1$ points and g at z_*, all under their common commitment. Every proof verifies by correctness. No polynomial of degree at most n fits these $n+2$ values since the first $n+1$ determine f, which disagrees with the last value. Thus their interpolation and the verifying openings give a degree-binding violation. □

Extended $(2, 2n{+}1)$-special soundness.

Lemma 6. *The compiled Boolean check is extended $(2, 2n{+}1)$-special sound for R_{bool} under the hardness of $R_{hd} = R_{\mathsf{bind}}$.*

Proof (Proof sketch.). Let T be an accepting $(2, 2n{+}1)$-tree with round-1 challenges $\alpha_1 \neq \alpha_2$. In the b-th subtree, the query challenges $z_{b,0}, \ldots, z_{b,2n}$ are pairwise distinct, with opened evaluations $\bar{f}_{bj}, \bar{g}_{bj}, \bar{q}_{bj}$. For each $b \in \{1, 2\}$, the extractor interpolates $\hat{f}_b$, $\hat{g}_b$ and q_b from the $2n+1$ openings of the b-th subtree. If any interpolated polynomial has degree larger than n, the corresponding openings form a degree-binding violation. If $\mathsf{Com}(\mathsf{ck}, \hat{f}_b) \neq C_f$, $\mathsf{Com}(\mathsf{ck}, \hat{g}_b) \neq C_g$, or $\mathsf{Com}(\mathsf{ck}, q_b) \neq C_{q_b}$, the openings form an interpolation-binding violation. In the remaining case, $\hat{f}_b$ and $\hat{g}_b$ have degree at most n and open C_f and C_g, which assures the first two clauses of R_{bool}. Moreover, $\hat{f}_1 = \hat{f}_2$ and $\hat{g}_1 = \hat{g}_2$, since two distinct polynomial openings of one commitment violate plain binding and hence degree binding. Write $\hat{f}$ and $\hat{g}$ for the common values. The acceptance conditions show that, for $b \in \{1, 2\}$ and all j,

$$V\big(\hat{f}(z_{bj})\big) + \alpha_b V\big(\hat{g}(z_{bj})\big) = q_b(z_{bj})\, Z_H(z_{bj}),$$

which is an equality of two polynomials of degree at most $2n$ at $2n+1$ distinct points. Hence this gives us a polynomial identity. For every $a \in H$ we have $Z_H(a) = 0$, so both identities give $V(\hat{f}(a)) + \alpha_b V(\hat{g}(a)) = 0$ for $b \in \{1, 2\}$. Subtracting the two equations and using $\alpha_1 \neq \alpha_2$ yields $V(\hat{g}(a)) = 0$, and then $V(\hat{f}(a)) = 0$. Hence $\hat{f}(a), \hat{g}(a) \in \{0, 1\}$ for every $a \in H$, and the extractor outputs the valid witness $(\hat{f}, \hat{g})$. □

Critical-round $(1, 2n{+}1)$-special soundness.

Lemma 7. *The compiled Boolean check is plain $(1, 2n{+}1)$-critical-round special sound (Definition 14) for R_{bool} at round $i^*_{\mathsf{ss}} = 2$.*

Proof (Proof sketch.). Given a $(1, 2n{+}1)$-tree T^*, the critical extractor CrExt interpolates $\hat{f}$ and $\hat{g}$ from the $2n+1$ opened evaluations and outputs $(\hat{f}, \hat{g})$. This is a complete candidate witness, and since Com is deterministic it can even be checked against R_{bool} directly.

Suppose $R_{\mathsf{bool}}(x, (\hat{f}, \hat{g})) = 0$, and let $T \sqsupseteq T^*$ be any accepting $(2, 2n{+}1)$-tree. Since $k_2 = h_2 = 2n+1$, the query branches of T^* are exactly the query branches of T below one of its round-1 branches, so the interpolation of ExExt in that subtree reproduces the candidate verbatim. Run ExExt of Lemma 6 on T. If it violates degree binding or interpolation binding, we are done. Otherwise, it outputs a valid witness, and this witness equals the candidate: one subtree interpolates to the candidate itself, and the other subtree agrees with it by plain binding, implied by degree binding (Lemma 5). This contradicts the invalidity of the candidate. Hence every such T yields a solution to R_{bind}, as required by Definition 14. In summary, degree binding bounds the degrees of the interpolated polynomials and forces the two subtrees to agree, and interpolation binding ties the interpolations to the commitments in the statement. □

Proposition 1 (Random-challenge unique responses). *For the compiled Boolean check specified above and every PPT adversary A making at most Q calls to Ver in the RUR game of [RT25, Definition 5 and Fig. 1], there is an expected-polynomial-time adversary B against the explicit relation R_{bind} such that*

$$\mathsf{Adv}^{\mathrm{rur}}_{A,\mathrm{bool}}(\lambda) \leq \mathsf{Adv}^{\mathrm{bind}}_{B}(\lambda) + \frac{8Qn + Q}{p}.$$

Here the binding advantage is the probability of outputting either tagged violation defined above, in the same honestly generated commitment-key experiment. The RUR adversary chooses its statement after receiving this key and before its first verifier query.

Proof. Let $\rho := \mathsf{Adv}^{\mathrm{rur}}_{A,\mathrm{bool}}$. Simulate the RUR verifier as described after Corollary 3, using at most Q oracle queries. The original winning transcripts have the same α and first differ at C_q or at the last prover response. By Corollary 3, both complete to accepting $(1, 2n+1)$-trees with probability at least $\rho - 8Qn/p$. Interpolate polynomials f_b, g_b, q_b from the $2n+1$ leaves of tree $b \in \{0,1\}$. If a degree exceeds n, output a degree-binding violation. If an interpolated polynomial's commitment is not the corresponding $C_f, C_g, C_q^{(b)}$, output an interpolation-binding violation. Otherwise f_0, f_1 open C_f, and g_0, g_1 open C_g. Any disagreement yields a degree-binding violation by Lemma 5. In the remaining case we write f, g for the common polynomials.

Acceptance at $2n+1$ distinct points gives, for each b, the degree-at-most-$2n$ identity

$$f(X)(f(X)-1) + \alpha g(X)(g(X)-1) = q_b(X)Z_H(X).$$

As $Z_H \neq 0$, the two identities imply $q_0 = q_1$. Determinism of Com now gives $C_q^{(0)} = C_q^{(1)}$, excluding a first divergence at C_q. In the other case the original transcripts also have the same z. Since the trees retain their original leaves, the common polynomials f, g, q force all three evaluation values in these responses to agree.

It remains to compare the opening proofs themselves. Write the standard KZG verification equation in additive notation as

$$e(C - vG_1, G_2) = e(W, (\tau - z)G_2),$$

where τ is the setup secret and the groups have prime order p. For fixed C, v, z with $z \neq \tau$, nondegeneracy makes the map $W \mapsto e(W, (\tau - z)G_2)$ injective. Hence each of the three proof group elements is unique, and canonical encodings make the entire response unique. Conditioned on any generated setup and the previous oracle history, a fresh second-round answer equals the fixed τ with probability $1/p$. The original adversary makes at most Q queries, so the probability that any of its second-round answers equals τ is at most Q/p. This charge concerns only the original run. The reduction does not need to know τ or detect this event. It simply outputs a binding witness whenever one of the preceding checks finds one, and otherwise fails. Outside this event every joint completion yields a binding violation, giving success probability at least $\rho - (8Qn + Q)/p$. □

Application to the multi-round Fischlin transform. Proposition 1 and Theorem 8 establish the straight-line knowledge guarantee for this Boolean check under the stated binding assumptions, when the polynomial-over-field-size terms are negligible. Here $L = 2n$, and the batching challenge α needs no probing. The concrete error contains the binding advantage in Proposition 1 and $(8Qn + Q)/p$, in addition to the terms of Theorem 8 other than its RUR term. For the asymptotic claim, strict-PPT binding suffices. A uniformly expected polynomial time reduction with non-negligible success can be truncated at a sufficiently large polynomial time, losing less than half that success probability by Markov's inequality. This does not assert the same concrete bound for a fixed strict-time budget.

Discussion. In the plain setting, the Fiat–Shamir knowledge error of Theorem 1 is governed by the branching numbers alone and equals $2(Q+1)(2n+1)/|\mathbb{F}|$. This matches the tightness goal of [KLPS26]. The price of this structure is witness-sized branching at the query round. Hiding is obtained by randomizing the committed polynomials rather than the commitments, and the hardness assumptions are pairing-based. The compiler of [KLPS26] applies to any special-sound polynomial IOP, and the argument above places the critical round at the query round whenever the polynomial IOP witness is determined by the query answers. Well-known zk-SNARKs of this shape include Plonk, Marlin, Lunar, and Vampire [KLPS26].

Plonk. The same mechanism yields a critical-round statement for Plonk [GWC19] with the KZG scheme. Lipmaa, Parisella and Siim [LPS25] prove that the five-round variant of Plonk is computationally $(3n{+}1, 3n{+}1, 3, 4n{+}21, 6)$-special sound for circuits with at most n gates, under falsifiable assumptions. The five challenges are the permutation challenges β and γ, the batching challenge α, the evaluation point $\mathfrak{z}$ and the opening batching challenge v. Their proof starts with a subtree analysis for fixed β, γ and α, which recovers the committed polynomials by separating the six batched openings along the v-branches and interpolating along the $4n{+}21$ evaluation branches. In our terminology, that subtree is a critical tree, and the branches of the first three rounds enter only the validity argument.

Proposition 2. *The five-round Plonk described in [LPS25] is plain critical-round special sound (Definition 14) at round $i^*_{\mathsf{ss}} = 4$ for circuit satisfiability, with critical tree shape $(1, 1, 1, 4n{+}21, 6)$.*

Proof (Proof sketch.). The critical extractor runs the subtree extractor of [LPS25, Section 5.4] and outputs the evaluations of the interpolated wire polynomials over the constraint domain. The blinding terms vanish on that domain, so these evaluations are the wire values. This candidate witness is complete, and it can be checked against the circuit directly since the statement of Plonk is fixed before the interaction. Suppose the candidate is invalid and let T be any accepting extension with branching $(3n{+}1, 3n{+}1, 3, 4n{+}21, 6)$. The wire commitments precede every challenge. As in Lemma 7, the witness computed by the extractor of [LPS25] from T therefore agrees with the candidate by interpolation, or some pair of verifying openings violates a binding notion. In the first case the candidate is valid, a contradiction. In the second case $\mathsf{ExExt}(T)$ outputs a solution for R_{hd}. □

Fischlin applicability. This Plonk variant does not satisfy the RUR hypothesis of Theorem 8, so that theorem and its Fischlin efficiency guarantee do not apply to it.

B.4 Sumcheck Sub-protocol of Spartan

We revisit the sumcheck sub-protocol of Spartan as analyzed by Dao and Grubbs [DG23]. Their Lemma 5.5 establishes a tree-extraction guarantee for this sub-protocol using the *partition predicate* of their Definition 4.2. We show that the round polynomial can already be extracted from a $(1, 1, 2)$-tree, while a $(1, 2, 2)$-tree certifies its consistency with the claim commitments, or yields a discrete-logarithm relation. We further show that deferring the inner Σ-protocol challenges allows all round polynomials to be extracted by branching only at the last round.

The sumcheck sub-protocol. Let $\mathbb{F}$ be a prime field and let Com be a Pedersen commitment scheme whose binding hardness is captured by the discrete-logarithm relation DL (denoted as R_{hd} below). Fix an iteration index i, a degree bound $l_i \in \mathbb{N}$, and a previous claim $e_{i-1} \in \mathbb{F}$ together with a commitment $C_{e_{i-1}}$ to e_{i-1}. One iteration of the Spartan sumcheck sub-protocol is the following seven-move public-coin protocol with three challenge rounds:

1. (Prover) Compute a polynomial $p_i(X) \in \mathbb{F}[X]$ of degree at most l_i, pick randomness ρ_{p_i}, and send $a := C_{p_i} = \mathsf{Com}(\mathsf{pp}, p_i; \rho_{p_i})$, a commitment to the coefficient vector $p_i \in \mathbb{F}^{l_i+1}$.
2. (Verifier) Send $c_1 := r_i \leftarrow\!\!\$\ \mathbb{F}$.
3. (Prover) Compute $e_i := p_i(r_i)$, pick randomness ρ_{e_i}, and send $z_1 := C_{e_i} = \mathsf{Com}(\mathsf{pp}, e_i; \rho_{e_i})$.
4. (Verifier) Send $c_2 := (w_{i,1}, w_{i,2}) \leftarrow\!\!\$\ \mathbb{F}^2$.
5. Both Compute $\mathbf{a} = w_{i,1} \cdot (\mathbf{0}^{l_i} + \mathbf{1}^{l_i}) + w_{i,2} \cdot \mathbf{r}_i^{l_i}$, and $C_{u_i} = C_{e_{i-1}}^{w_{i,1}} \cdot C_{e_i}^{w_{i,2}}$.
   Prover additionally computes $u_i = w_{i,1} e_{i-1} + w_{i,2} e_i$, $\rho_{u_i} = w_{i,1} \rho_{e_{i-1}} + w_{i,2} \rho_{e_i}$.
6. (Both) Run the inner Σ-protocol $\mathsf{DotProdPf}$ on instance $(\mathsf{pp}, (C_{p_i}, C_{u_i}, \mathbf{a}))$ and witness $(p_i, \rho_{p_i}, u_i, \rho_{u_i})$, with prover commitment z_2, verifier challenge $c_3 \in \mathbb{F}$, and prover response z_3.

The verifier accepts iff $\mathsf{DotProdPf}$ accepts. The protocol has three challenge rounds, with challenge spaces $\mathcal{C}_1 = \mathbb{F}$, $\mathcal{C}_2 = \mathbb{F}^2$, $\mathcal{C}_3 = \mathbb{F}$.

Extraction guarantee. As noted in [DG23], the sumcheck sub-protocol is not strictly an interactive argument. Following their Lemma 5.5, we state the extraction guarantee for the round polynomials and claim commitments. The critical extractor outputs the round polynomial p_i. On an accepting extended tree satisfying $\boldsymbol{\Phi}$, ExExt either outputs a solution to $R_{hd} = \mathsf{DL}$, or extracts openings $(\widehat{e}_{i-1}, \widehat{\rho}_{i-1})$ and $(\widehat{e}_i, \widehat{\rho}_i)$ of $C_{e_{i-1}}$ and C_{e_i} such that

$$p_i(0) + p_i(1) = \widehat{e}_{i-1}, \qquad p_i(r_i) = \widehat{e}_i, \qquad \deg p_i \le l_i.$$

Here the claim values are extracted from the commitments. Linking these openings to a fixed Spartan statement requires the surrounding sub-protocols, as in the composition of [DG23, Lemma 5.6].

The partition predicate from Definition 4.2 of [DG23]. A partition of a challenge space $\mathcal{C}$ into blocks $\mathcal{C} = \mathcal{C}^{(1)} \sqcup \cdots \sqcup \mathcal{C}^{(n)}$ gives rise to a predicate that evaluates to 1 on a tuple of challenges iff they lie in pairwise distinct blocks. For round 2 here, we take the *linear-independence partition* of $\mathbb{F}^2 \setminus \{(0,0)\}$. Two pairs $(w_{i,1}, w_{i,2})$ and $(w'_{i,1}, w'_{i,2})$ lie in distinct blocks iff they are linearly independent over $\mathbb{F}$. We denote this predicate by $\Phi^{\mathrm{li}}_{2,\ell}$ for $\ell \in [2]$. Rounds 1 and 3 use the trivial *distinctness* predicate.

We instantiate the predicate system $\boldsymbol{\Phi}$ of Definition 12 on $(h_1, h_2, h_3) = (1, 2, 2)$-trees. The critical predicate Φ_{cr} on $(k_1, k_2, k_3) = (1, 1, 2)$-trees is the trivial one, since only round 3 branches and pairwise distinctness suffices. The critical round is $i^*_{\mathsf{ss}} = 3$.

Extraction from a $(1, 1, 2)$-critical tree.

Lemma 8. *Given an accepting $(1, 1, 2)$-tree T^* of the sumcheck sub-protocol of Figure 8 of [DG23],* CrExt *outputs a polynomial p_i of degree at most l_i. For every accepting $(1, 2, 2)$-tree $T \sqsupseteq T^*$ with $\boldsymbol{\Phi}(T) = 1$,* ExExt *either outputs a DL-solution, or claim openings satisfying the extraction guarantee above with the same p_i.*

Proof (Proof sketch.). The critical extractor CrExt is given a $(1, 1, 2)$-tree T^*, i.e., a single fixed round-1 challenge r_i and the round-2 challenge $(w_{i,1}, w_{i,2})$, branching only at the round-3 challenge $c_3 \in \mathbb{F}$ of $\mathsf{DotProdPf}$. The two leaves of T^* are two accepting transcripts of $\mathsf{DotProdPf}$ on the common instance $(\mathsf{pp}, (C_{p_i}, C_{u_i}, \mathbf{a}))$ that differ only at c_3.

As $\mathsf{DotProdPf}$ is a 2-special sound Σ-protocol for the dot-product relation [DG23, Lemma 5.3], its 2-tree extractor outputs $(p_i, \rho_{p_i}, u_i, \rho_{u_i})$ such that

$$C_{p_i} = \mathsf{Com}(\mathsf{pp}, p_i; \rho_{p_i}), \qquad C_{u_i} = \mathsf{Com}(\mathsf{pp}, u_i; \rho_{u_i}), \qquad \langle p_i, \mathbf{a} \rangle = u_i,$$

and by definition of $\mathbf{a}$, we have $\langle p_i, \mathbf{a} \rangle = w_{i,1}(p_i(0) + p_i(1)) + w_{i,2}\, p_i(r_i)$. The critical extractor outputs p_i as the candidate witness.

Let $T \sqsupseteq T^*$ be any accepting $(1, 2, 2)$-tree with $\boldsymbol{\Phi}(T) = 1$. Since $k_3 = h_3 = 2$, the two leaves of T^* form the round-3 pair of T under one of its round-2 branches. The extraction of Lemma 9 under that branch therefore returns the same p_i. Hence either $\mathsf{ExExt}(T)$ outputs a DL-solution, or it outputs claim openings with values $\widehat{e}_{i-1} = p_i(0) + p_i(1)$ and $\widehat{e}_i = p_i(r_i)$, as stated. □

Extraction from a $(1, 2, 2)$-extended tree with the partition predicate.

Lemma 9. *Given an accepting $(1, 2, 2)$-tree satisfying $\boldsymbol{\Phi}$,* ExExt *outputs a polynomial p_i of degree at most l_i and openings of $C_{e_{i-1}}$ and C_{e_i} with values $\widehat{e}_{i-1} = p_i(0) + p_i(1)$ and $\widehat{e}_i = p_i(r_i)$, or a solution to $R_{hd} = \mathsf{DL}$. Here $\boldsymbol{\Phi}$ imposes $\Phi^{\mathrm{li}}_{2,1}$ and $\Phi^{\mathrm{li}}_{2,2}$ at round 2.*

Proof (Proof sketch.). Let T be a $(1, 2, 2)$-tree of accepting transcripts with $\boldsymbol{\Phi}(T) = 1$. By the partition predicate, the two round-2 challenges $(w_{i,1}, w_{i,2})$ and $(w'_{i,1}, w'_{i,2})$ are linearly independent over $\mathbb{F}$.

T is partitioned into two $(1, 1, 2)$-subtrees T^*, $T^{*'}$, one per round-2 branch. Given as input $(p_i, \rho_{p_i}, u_i, \rho_{u_i})$ from T^* and $(p'_i, \rho'_{p_i}, u'_i, \rho'_{u_i})$ from $T^{*'}$, the extended extractor ExExt proceeds as follows:

1. If $(p_i, \rho_{p_i}) \neq (p'_i, \rho'_{p_i})$, ExExt outputs the pair as a R_{hd}-solution: both opening the same commitment C_{p_i} to different values give a non-trivial solution to the discrete-log relation.
2. Otherwise, $p_i = p'_i$. The dot-product equations $\langle p_i, \mathbf{a} \rangle = u_i$ give the following equation:

$$\begin{pmatrix} w_{i,1} & w_{i,2} \\ w'_{i,1} & w'_{i,2} \end{pmatrix} \begin{pmatrix} p_i(0) + p_i(1) \\ p_i(r_i) \end{pmatrix} = \begin{pmatrix} u_i \\ u'_i \end{pmatrix},$$

which is invertible by the partition predicate. Applying the inverse to this equation, and correspondingly to the homomorphic equations defining C_{u_i} and C'_{u_i}, opens $C_{e_{i-1}}$ to the value $p_i(0) + p_i(1)$ and C_{e_i} to the value $p_i(r_i)$. The randomness of the two openings is given by the same linear combinations of ρ_{u_i} and ρ'_{u_i}. The extractor outputs p_i and these openings, writing $\widehat{e}_{i-1} := p_i(0) + p_i(1)$ and $\widehat{e}_i := p_i(r_i)$ for the extracted claim values. The extracted p_i is a coefficient vector of length $l_i + 1$, so $\deg p_i \leq l_i$.

In every case ExExt outputs either the polynomial and claim openings, or a solution s for R_{hd}. □

Failure density of the round-2 predicate. The linear-independence partition of $\mathbb{F}^2 \setminus \{(0,0)\}$ has $|\mathbb{F}|+1$ blocks of size $|\mathbb{F}|-1$. Given any fixed first challenge $(w_{i,1}, w_{i,2}) \neq (0,0)$, the set of round-2 challenges that violate $\Phi^{\mathsf{li}}_{2,2}$ is the block containing it together with $(0,0)$, of size $|\mathbb{F}|$. Hence the failure density of $\Phi^{\mathsf{li}}_{2,2}$ is

$$p^{\mathsf{chal}}_{2,2} \leq \frac{|\mathbb{F}|}{|\mathbb{F}|^2} = \frac{1}{|\mathbb{F}|}.$$

The predicate $\Phi^{\mathsf{li}}_{2,1}$ on the first round-2 challenge only requires $(w_{i,1}, w_{i,2}) \neq (0,0)$, with failure density $p^{\mathsf{chal}}_{2,1} = 1/|\mathbb{F}|^2$.

Full Sum-check with iterations. Let μ be the number of the variables in the polynomial p. The full sum-check repeats μ times of the above iteration, giving a $(1,2,2)^\mu$-extended tree and a $(1,1,2)^\mu$-critical tree. Here C_{e_i} serves as the input claim commitment of iteration $i+1$. Along a fixed path, the extended extractor recovers openings of these commitments with values $\widehat{e}_0, \ldots, \widehat{e}_\mu$ satisfying

$$p_i(0) + p_i(1) = \widehat{e}_{i-1}, \qquad p_i(r_i) = \widehat{e}_i \qquad (i \in [\mu]),$$

or outputs a solution to $R_{hd} = \mathsf{DL}$. In particular, equal openings of each intermediate commitment give $p_{i+1}(0) + p_{i+1}(1) = p_i(r_i)$.

One can in fact push the round 3 in each iteration all the way to the very end of the protocol by a following trick: in round 3 of each iteration i, the prover only sends the first message of DotProdPf and then immediately proceeds to iteration $i+1$. After all μ iterations have produced their DotProdPf commitments, the verifier sends a single vector of μ Σ-protocol challenges $(c_3^{(1)}, \ldots, c_3^{(\mu)})$ in one move, and the prover responds to all μ responses of DotProdPf in one final move.

The resulting protocol has the tree-extraction guarantee of Lemma 10, with extended shape $((1,2)^\mu, 2)$ and critical shape $((1,1)^\mu, 2)$, branching only at the last round. The per-iteration linear-independence partition predicates $\Phi^{\mathsf{li}}_{2,1}$ and $\Phi^{\mathsf{li}}_{2,2}$ still apply at the $(w_{i,1}, w_{i,2})$ rounds in $\boldsymbol{\Phi}'$.

As the final round now carries a vector of challenges $\vec{c}_3 \in \mathbb{F}^\mu$, the final round predicate must be updated: instead of asking $\vec{c}_3 \neq \vec{c}_3{}'$ (which could differ in only a single coordinate), one requires *pairwise distinctness in every coordinate*, i.e. $c_3^{(i)} \neq c_3{}'^{(i)}$ for all $i \in [\mu]$. This is exactly needed for each per-iteration DotProdPf 2-special sound extractor to work independently. The new predicate is a generalized challenge predicate with failure density at most $\mu/|\mathbb{F}|$ by a union bound.

Lemma 10. *Given an accepting $((1,1)^\mu, 2)$-tree T^* satisfying Φ'_{cr}, the critical extractor outputs round polynomials $(p_1, \ldots, p_\mu)$ with $\deg p_i \leq l_i$. For every accepting $((1,2)^\mu, 2)$-tree $T \sqsupseteq T^*$ satisfying $\boldsymbol{\Phi}'$, the extended extractor either outputs a DL-solution, or openings of the claim commitments along the common prefix of T^*, with values $\widehat{e}_0, \ldots, \widehat{e}_\mu$ such that*

$$p_i(0) + p_i(1) = \widehat{e}_{i-1}, \qquad p_i(r_i) = \widehat{e}_i \qquad (i \in [\mu]).$$

Proof (Proof sketch.). The two final challenge vectors differ in every coordinate, so the DotProdPf extractors recover $(p_1, \ldots, p_\mu)$ from T^*. On T, the two linearly independent weight vectors at each iteration give openings of its two claim commitments as in Lemma 9. The recovered polynomials agree with those of T^*, or the two openings of some C_{p_i} give a DL-solution. Likewise, the openings obtained from adjacent iterations for each intermediate C_{e_i} either agree, yielding the stated equations, or give a DL-solution. □

B.5 Protogalaxy

We close our series of examples of critical-round special-sound protocols with a folding scheme. A (multi-) folding scheme is a proof system that combines k witnesses of a relation R and a witness of an accumulator relation R^* into a single witness of an accumulator relation R^*. The prover knows the witnesses in the clear, whereas the verifier only needs commitments to those witnesses. Folding schemes are modelled as reductions of knowledge: for a folding scheme $R^k \times R^* \to R^*$, if the proof verifies *and* the output witness is indeed in the accumulator relation R^*, then one can extract the original witnesses and the initial accumulator.

Protogalaxy [EG23] is an algebraic folding scheme that linearly combines the k base witnesses with the accumulator using Lagrange polynomials. In the knowledge-soundness proof, the extractor

obtains the $k+1$ folded witnesses from a tree that grows with the size of the folded instance (e.g. a R1CS instance). Here, the extraction itself only uses the last round. In particular, we note that a critical subtree with a single $k+1$ branching at the final challenge already determines all folded witnesses uniquely. The remaining branches serve to certify that the extracted witnesses are valid, or else to break the binding of the commitment scheme. We will show that Protogalaxy is plain CRSS at its last round.

Setting. We recall some notation from Protogalaxy. Let $\mathbb{F}$ be a prime field and let $\mathbb{G}$ be a group of order $|\mathbb{F}|$. Let $\mathsf{Com}(\mathsf{ck}, \cdot) : \mathbb{F}^M \to \mathbb{G}$ be a deterministic, additively homomorphic and binding commitment scheme. The folding scheme takes, as instances, transcripts of a committed special-sound protocol following [BC23], with a degree-d verification map $f : \mathbb{F}^M \to \mathbb{F}^n$. To avoid notational clutter, we assume that all folded transcripts come from a single-round special-sound protocol. This is to avoid introducing hash-consistency, but our analysis extends easily to this case. Hence, our witnesses are vectors $\omega \in \mathbb{F}^M$ and they belong to the folding relation if $f(\omega) = 0^n$. As an example, one can think of $\omega = (\vec{x}, \vec{w}, 1)$ as the extended witness of an R1CS relation, and of f as the R1CS matrices A, B, C.

Let $n = 2^t$ and, for $i \in [n]$, let $S_i \subseteq \{0, \ldots, t-1\}$ be the index set with $i - 1 = \sum_{\ell \in S_i} 2^\ell$ and $\mathsf{pow}_i(X_0, \ldots, X_{t-1}) := \prod_{\ell \in S_i} X_\ell$ the corresponding multilinear monomial. Fix $1 \leq d, k < n$ with $k+1$ a power of two, let $H \subset \mathbb{F}^*$ be a multiplicative subgroup of order $k+1$ with Lagrange basis $L_0, \ldots, L_k$ and vanishing polynomial $Z(X) := \prod_{a \in H}(X - a)$ of degree $k+1$. Assume $n, t+1, dk+1 \leq p := |\mathbb{F}|$ and polynomially bounded tree size $n(t+1)(dk+1)$.

Folding Relations. Protogalaxy involves two folding relations. The first is the base relation $R := R_{f,\mathsf{Com}}$, consisting of the pairs (ϕ, ω) such that $\phi = \mathsf{Com}(\mathsf{ck}, \omega)$ and $f(\omega) = 0^n$. The second is its *randomized relaxed* version R_{rand}, consisting of the pairs $\big((\phi, \boldsymbol{\beta}, e), \omega\big)$ with $\phi = \mathsf{Com}(\mathsf{ck}, \omega)$, $\boldsymbol{\beta} \in \mathbb{F}^t$, $e \in \mathbb{F}$ and

$$\textstyle\sum_{i \in [n]} \mathsf{pow}_i(\boldsymbol{\beta})\, f_i(\omega) = e,$$

where f_i denotes the i-th output coordinate of f.

Folding Protocol. Formally, Protogalaxy is a reduction of knowledge: given an accepting proof and a valid folded instance (i.e., a folded witness ω^* that belongs to R_{rand}), the extractor must be able to obtain the original witnesses $(\omega_1, \ldots, \omega_k)$. On input $\big((\phi, \boldsymbol{\beta}, e), (\phi_1, \ldots, \phi_k)\big)$, with the prover additionally knowing ω and $\omega_1, \ldots, \omega_k$, the protocol proceeds as follows.

1. (Verifier) Send $c_1 := \delta \leftarrow\!\$\, \mathbb{F}$. Both parties set $\boldsymbol{\delta} := (\delta, \delta^2, \delta^4, \ldots, \delta^{2^{t-1}}) \in \mathbb{F}^t$.
2. (Prover) Send $z_1 := (F_1, \ldots, F_t)$, the non-constant coefficients of

   $$F(X) := \textstyle\sum_{i \in [n]} \mathsf{pow}_i(\boldsymbol{\beta} + X\boldsymbol{\delta})\, f_i(\omega),$$

   a polynomial of degree at most t.
3. (Verifier) Send $c_2 := \alpha \leftarrow\!\$\, \mathbb{F}$. Both parties compute $F(\alpha) := e + \sum_{i \in [t]} F_i \alpha^i$ and $\boldsymbol{\beta}^* := \boldsymbol{\beta} + \alpha\boldsymbol{\delta}$.
4. (Prover) Send $z_2 := K(X)$, the quotient in $G(X) = F(\alpha)L_0(X) + Z(X)K(X)$, where

   $$G(X) := \textstyle\sum_{i \in [n]} \mathsf{pow}_i(\boldsymbol{\beta}^*)\, f_i\big(L_0(X)\omega + \sum_{j \in [k]} L_j(X)\omega_j\big)$$

   has degree at most dk.
5. (Verifier) Send $c_3 := \gamma \leftarrow\!\$\, \mathbb{F}$. Both parties compute $e^* := F(\alpha)L_0(\gamma) + Z(\gamma)K(\gamma)$ and $\phi^* := \sum_{j=0}^{k} L_j(\gamma)\phi_j$, writing $\phi_0 := \phi$.
6. (Prover) Send $z_3 := \omega^* = \sum_{j=0}^{k} L_j(\gamma)\omega_j$, the folded witness.

The verifier accepts if $\big((\phi^*, \boldsymbol{\beta}^*, e^*), \omega^*\big) \in R_{\mathsf{rand}}$. The message K has degree at most $dk - k - 1$ and is encoded by the fixed coefficient vector for this bound. When the bound is negative, only the zero polynomial encoded by the empty vector is allowed. The verifier rejects messages outside this format. This degree restriction also supplies the bound on ZK used in Lemma 13 below.

Statement, witness, and hardness relation. Following Definition 12, the statement splits into an instance for the folded relation and a hard instance that corresponds to commitment binding. We take

$$x := \big((\phi, \boldsymbol{\beta}, e), \phi_1, \ldots, \phi_k\big), \qquad w := (\omega_0, \omega_1, \ldots, \omega_k), \qquad R^{\mathsf{fold}} := R_{\mathsf{rand}} \times R^k,$$

writing $\omega_0 := \omega$, and as the hard instance the commitment key, $y := \mathsf{ck}$, with $R_{hd}(y, s) = 1$ iff $s = (u, u')$ satisfies $u \neq u'$ and $\mathsf{Com}(\mathsf{ck}, u) = \mathsf{Com}(\mathsf{ck}, u')$. We note that this is exactly the extended relation of [EG23, Lemma 3.2], which reduces knowledge soundness for R^{fold} to special soundness for R^{fold} extended by commitment binding.

Extended $(n, t{+}1, dk{+}1)$-special soundness. Before showing extended special soundness, we require two lemmas. The first one is borrowed directly from [EG23, Lemma 4.2]; the second one is necessary to argue that the extractor never fails.

Lemma 11. *Let $g : \mathbb{F}^M \to \mathbb{F}$ be a polynomial map and let $u_0, \ldots, u_k \in \mathbb{F}^M$. Then*

$$g\Big(\textstyle\sum_{j=0}^{k} L_j(X)\, u_j\Big) \;\equiv\; \sum_{j=0}^{k} L_j(X)\, g(u_j) \pmod{Z(X)}.$$

Lemma 12. *For pairwise distinct $\gamma_0, \ldots, \gamma_k \in \mathbb{F}$, the matrix $A := \big(L_j(\gamma_\ell)\big)_{\ell, j \in \{0,\ldots,k\}}$ is invertible.*

Proof. We show that A is full-rank. Let $v \in \mathbb{F}^{k+1}$ with $Av = 0$, that is, $\sum_j v_j L_j(\gamma_\ell) = 0$ for every ℓ. Then the polynomial $q(X) := \sum_j v_j L_j(X)$, of degree at most k, vanishes at the $k+1$ distinct points $\gamma_0, \ldots, \gamma_k$, so $q \equiv 0$. As $\{L_j\}_{j=0}^k$ is a basis of the space of polynomials of degree at most k, it follows that all $v_j = 0$.

Lemma 13. *Protogalaxy is extended $(n, t{+}1, dk{+}1)$-special sound for R^{fold} under the hardness of R_{hd}.*

Proof (Sketch). What follows is essentially a rearrangement of the knowledge-soundness proof in [EG23, Theorem 4.3]. Let T be a $(n, t{+}1, dk{+}1)$ accepting tree, where we index the branches by $a \in [n]$, $b \in [t+1]$, $c \in [dk+1]$, with challenges δ_a, $\alpha_{a,b}$, $\gamma_{a,b,c}$, respectively.

We construct extractor ExExt as follows.

1. Interpolate the witnesses. Fix the branch $(a, b) = (1, 1)$ and any $k{+}1$ of its leaves, say those with $c \in [k{+}1]$, and abbreviate $\gamma_\ell := \gamma_{1,1,\ell}$ and $\omega^*_\ell := \omega^*_{1,1,\ell}$. Then, let $(\omega'_0, \ldots, \omega'_k)^\top := A^{-1}(\omega^*_1, \ldots, \omega^*_{k+1})^\top$ with A being the matrix of Lagrange coefficients as in Lemma 12. Since all of these are accepting transcripts, we have that $\mathsf{Com}(\mathsf{ck}, \omega^*_\ell) = \phi^*_\ell = \sum_j L_j(\gamma_\ell)\phi_j$, and applying the same linear combinations yields $\mathsf{Com}(\mathsf{ck}, \omega'_j) = \phi_j$ for every $j \in \{0, \ldots, k\}$.

2. Certify consistency or output a collision. For *every* remaining leaf of T, with round-3 challenge γ and message ω^*, the extractor tests whether $\omega^* = \sum_j L_j(\gamma)\omega'_j$. Note that both sides are pre-images of $\phi^* = \sum_j L_j(\gamma)\phi_j$: the left one by acceptance (as it is a tree leaf), the right one by the previous step. Therefore, if the test fails, ExExt outputs a break of commitment binding and stops.

3. Verify that the extracted ω'_j satisfy relation R. This is essentially the argument of the original knowledge-soundness proof, so we only sketch it. First, the extractor uses the challenges at rounds 1 and 2 to make sure that, by establishing several polynomial equalities, $(\phi_j, \omega'_j) \in R$ for every $j \in [k]$. Second, it asserts that the initial accumulator $\big((\phi, \boldsymbol{\beta}, e), \omega'_0\big) \in R_{\mathsf{rand}}$ as well.

For the first part, fix a subtree (a, b) and use step 2 to substitute $\omega^*_{a,b,c} = \sum_j L_j(\gamma_{a,b,c})\omega'_j$ in the acceptance equations of its $dk + 1$ leaves. Both sides have degree at most dk, so this yields the polynomial identity

$$\sum_i \mathsf{pow}_i(\boldsymbol{\beta}^*_{a,b}) f_i\Big(\sum_j L_j(X)\omega'_j\Big) = F^{(a)}(\alpha_{a,b})L_0(X) + Z(X)K^{(a,b)}(X).$$

Reducing modulo $Z(X)$ via Lemma 11 and comparing Lagrange coefficients gives

$$\textstyle\sum_i \mathsf{pow}_i(\boldsymbol{\beta}^*_{a,b}) f_i(\omega'_0) = F^{(a)}(\alpha_{a,b}), \qquad \sum_i \mathsf{pow}_i(\boldsymbol{\beta}^*_{a,b}) f_i(\omega'_j) = 0 \quad (j \in [k]). \tag{6}$$

Fix $j \in [k]$. The second identity says that the polynomial $\sum_i \mathsf{pow}_i\big(\boldsymbol{\beta} + Y_1 \cdot (Y_2, Y_2^2, \ldots, Y_2^{2^{t-1}})\big) f_i(\omega'_j)$, of degree at most t in Y_1 and at most $n - 1$ in Y_2, vanishes on the grid $\{(\alpha_{a,b}, \delta_a)\}_{a \in [n], b \in [t+1]}$ and

hence it is identically zero. Since the monomials pow_i are linearly independent, $f_i(\omega'_j) = 0$ for all $i \in [n]$, i.e. $(\phi_j, \omega'_j) \in R$.

For the second part, for a fixed a the first identity gives $F^{(a)}(X) = \sum_i \mathsf{pow}_i(\boldsymbol{\beta} + X\boldsymbol{\delta}_a) f_i(\omega'_0)$. Comparing constant coefficients then yields $\sum_i \mathsf{pow}_i(\boldsymbol{\beta}) f_i(\omega'_0) = e$, that is, $\big((\phi, \boldsymbol{\beta}, e), \omega'_0\big) \in R_{\mathsf{rand}}$.

In every case ExExt outputs a witness $w = (\omega'_0, \ldots, \omega'_k)$ with $R^{\mathsf{fold}}(x, w) = 1$ or a solution s with $R_{hd}(y, s) = 1$. It is deterministic and runs in time $\mathsf{poly}(\lambda)$, since the tree has $n(t+1)(dk+1) = \mathsf{poly}(\lambda)$ leaves. □

Critical-round $(1, 1, k{+}1)$-special soundness.

Lemma 14. *Protogalaxy is plain $(1, 1, k{+}1)$-critical-round special sound (Definition 14) for R^{fold} at round $i^*_{\mathsf{ss}} = 3$.*

Proof (Sketch). A critical tree T^* consists of $k + 1$ accepting transcripts that share the prefix $(\delta, z_1, \alpha, z_2)$ and differ only in the last challenges γ_i. Let $\omega^*_0, \ldots, \omega^*_k$ be the final folded witnesses. Define

$$\mathsf{CrExt}(x, T^*): \quad (\omega_0, \ldots, \omega_k)^\top := A^{-1} (\omega^*_0, \ldots, \omega^*_k)^\top, \qquad A = \big(L_j(\gamma_\ell)\big)_{\ell, j},$$

which is well-defined by Lemma 12, and output $w := (\omega_0, \ldots, \omega_k)$.

We proceed by contradiction. Suppose $R^{\mathsf{fold}}(x, w) = 0$ and let $T \sqsupseteq T^*$ be any accepting $(n, t{+}1, dk{+}1)$-tree. Then, run $\mathsf{ExExt}(T)$ of Lemma 13. Either it stops in step 2 with a collision, and we are done, or it outputs $w' = (\omega'_0, \ldots, \omega'_k)$. In the latter case, ExExt has verified $\omega^* = \sum_j L_j(\gamma)\omega'_j$ at every leaf of T. Since the $k + 1$ leaves of T^* are leaves of T, it holds that $A\,(\omega'_0, \ldots, \omega'_k)^\top = (\omega^*_0, \ldots, \omega^*_k)^\top$. Observe that the vector w' solves the same system that defines the output of CrExt, but $w' = w$ and $R^{\mathsf{fold}}(x, w') = 0$ by assumption. This contradicts Lemma 13. Therefore, $\mathsf{ExExt}(T)$ outputs a solution to R_{hd} for every such T. □

Proposition 3 (Random-challenge unique responses). *For the witness-revealing protocol specified above and every PPT adversary A making at most Q calls to Ver in the RUR game of [RT25, Definition 5 and Fig. 1], there is a binding adversary B for Com that runs in expected polynomial time and satisfies*

$$\mathsf{Adv}^{\mathrm{rur}}_{A,\mathrm{PG}}(\lambda) \le \mathsf{Adv}^{\mathrm{bind}}_{B,\mathsf{Com}}(\lambda) + \frac{4Q(t + dk)}{p}.$$

Proof. Let $D := dk$ and $\rho := \mathsf{Adv}^{\mathrm{rur}}_{A,\mathrm{PG}}$. Generate the commitment key honestly in both experiments. Simulate the RUR verifier as described after Corollary 3. The adversary fixes its statement before its verifier queries. Apply Corollary 3 to complete both transcripts of the original winning pair to shape $(1, t{+}1, D{+}1)$. Both completions succeed with probability at least $\rho - 4Q(t{+}D)/p$. The initial prover message is empty, so the first differing response is F, K, or ω^*, and the two original transcripts have the same first challenge δ.

From any one α-branch of the first completed tree, take $k + 1$ distinct γ-leaves and recover $\omega_0, \ldots, \omega_k$ using the inverse matrix of Lemma 12. Acceptance and homomorphism give $\mathsf{Com}(\mathsf{ck}, \omega_j) = \phi_j$ for every j. At every leaf of both trees, compare the response ω^* with $\widehat{\omega}(\gamma) := \sum_j L_j(\gamma)\omega_j$. If they differ, they have the same commitment ϕ^*, and B outputs this binding violation. Otherwise all leaves of both trees are determined by these common vectors.

For any fixed α-branch, put

$$G_\alpha(X) := \sum_i \mathsf{pow}_i(\boldsymbol{\beta} + \alpha\boldsymbol{\delta}) f_i\big(\widehat{\omega}(X)\big).$$

The $D + 1$ accepting leaves and the degree bound on K imply the polynomial identity

$$G_\alpha(X) = F(\alpha)L_0(X) + Z(X)K(X), \tag{7}$$

since both sides have degree at most D. Evaluating at the interpolation node of L_0 gives $F(\alpha) = \sum_i \mathsf{pow}_i(\boldsymbol{\beta} + \alpha\boldsymbol{\delta}) f_i(\omega_0)$. There are $t + 1$ distinct α-branches in each tree, and both sides have degree at most t in α. Thus the two trees have the same polynomial

$$F(Y) = \sum_i \mathsf{pow}_i(\boldsymbol{\beta} + Y\boldsymbol{\delta}) f_i(\omega_0).$$

This excludes a first divergence at F. If it occurs at K, the original transcripts have the same α. (7) and $Z \neq 0$ force their K's to agree. If it occurs at ω^*, the original γ's also agree, and both responses equal $\widehat{\omega}(\gamma)$. Canonical encodings exclude distinct representations of equal messages. Therefore each joint completion yields a binding violation, proving the bound. □

Discussion. All challenge predicates are trivial, so the Fiat–Shamir knowledge error of Theorem 1 is

$$\kappa_{ex} \;=\; \frac{2(Q+1)\big((n-1)+t+dk\big)}{|\mathbb{F}|}, \qquad \text{with} \qquad \theta \;=\; \prod_{i\in[3]} \Big(1-\frac{h_i-1}{|\mathbb{F}|}\Big).$$

Application to the multi-round Fischlin transform. Proposition 3 supplies the response-uniqueness bound for the RUR game of [RT25, Definition 5], as required by Theorem 8. Thus the witness-revealing protocol above has the resulting straight-line knowledge guarantee under commitment binding, for polynomial parameters and negligible field-size error terms. The concrete bound charges the binding advantage of the RUR reduction in addition to the hardness term in Theorem 8. The expected-time reductions can be truncated at a sufficiently large polynomial cutoff: a non-negligible success probability and a uniform polynomial expected-time bound would otherwise give a strict-PPT binding attack by Markov's inequality. This asymptotic implication does not preserve a fixed concrete running-time budget.

Protogalaxy is a case where the CRSS variant of the multi-round Fischlin transform in Section 6 makes a substantial difference, since its special-soundness parameters $\vec{h} = (n, t{+}1, dk{+}1)$ grow with the folded instance (e.g., n is the number of R1CS rows) whereas its critical tree is tiny. With the original transform of [RT25], the quantity L of Theorem 8 (with $i^*_{\mathsf{ss}} = 1$) satisfies $L \geq (n{-}1)(t{+}1)(dk{+}1)$, so the proof-of-work threshold $2^\ell > L$ forces $\Omega(n \log n \cdot dk)$ random-oracle queries per repetition, each on a transcript containing $\omega^* \in \mathbb{F}^M$. Worse, since $\mathsf{increment}_{\mathbf{k}'}$ exhausts the last challenge set first, these probes traverse $\Omega(n \log n)$ distinct pairs (δ, α), each forcing a recomputation of $K(X)$, i.e., $dk+1$ evaluations of f of cost $\Omega(n)$ each. The transformed prover would thus become quadratic in the instance size, whereas the interactive prover evaluates f only $\mathcal{O}(dk)$ times.

By contrast, with $\vec{k} = (1, 1, k{+}1)$ at $i^*_{\mathsf{ss}} = \mu = 3$ (Lemma 14) the modified transform sets $k'_1 = k'_2 = 1$, so the prover derives δ and α by Fiat–Shamir, computes $F(X)$ and $K(X)$ once, and probes only γ, each probe being one interpolation $\omega^* = \sum_j L_j(\gamma)\omega_j$ of cost $\mathcal{O}(kM)$ plus one hash. Moreover, $L = k$ independently of n, t and d, and no additional evaluation of f is needed. In effect, the transform collapses to Fischlin's original transform [Fis05] on the last round of Protogalaxy, viewed as a Σ-protocol with first message $(\delta, z_1, \alpha, z_2)$, challenge γ and response ω^*. The straight-line extractor merely collects $k+1$ accepting transcripts with a common prefix and distinct γ's from the random-oracle queries and runs CrExt, i.e., solves a $(k{+}1) \times (k{+}1)$ linear system; the $(n, t{+}1, dk{+}1)$-tree of Lemma 13 enters only the reduction to binding via Theorem 8. Finally, we note that $(1, 1, k{+}1)$ is optimal: ω^* is a single linear combination of the $k{+}1$ unknowns $\omega_0, \ldots, \omega_k$, so at least $k{+}1$ transcripts, and hence one branching round of width $k+1$, are unavoidable.

C Katz-Kolesnikov-Wang Framework for MPC-in-the-Head with Preprocessing

We recall the KKW framework that follows the MPCitH paradigm [IKOS07]. In this framework, the prover runs an MPC protocol that evaluates a boolean circuit C on an input w, commits to the views of all parties, and then opens all-but-one of these views to the verifier. The protocol relies on an XOR-based n-out-of-n secret sharing scheme; we denote shares of value a by $[\![a]\!]$. This MPC protocol, in the preprocessing model, is secure against semi-honest all-but-one corruptions. The protocol can be summarized as follows:

- For each wire α of circuit C, let $z_\alpha \in \{0,1\}$ be the wire value. Each party holds a share $[\![\lambda_\alpha]\!]$ of a random value λ_α along with a masked value $\hat{z}_\alpha := z_\alpha + \lambda_\alpha$.
- The shares $[\![\lambda_\alpha]\!]$ are generated independent with the witnness w in the *preprocessing phase*. A gate is defined by an input wire α, β, and an output wire γ. For an XOR gate parties can locally compute the share of output wire $[\![\lambda_{\alpha\oplus\beta}]\!] = [\![\lambda_\alpha]\!] \oplus [\![\lambda_\beta]\!]$, while for a multiplication gate, each party are given $[\![\lambda_{\alpha\cdot\beta}]\!]$ with the help of a auxiliary value that is given to n-th party and computed by $(\sum_{i=1}^n [\![\lambda_\alpha]\!]) \cdot (\sum_{i=1}^n [\![\lambda_\beta]\!]) - \sum_{i=1}^{n-1} [\![\lambda_{\alpha\cdot\beta}]\!]_i$.

– The set of shares $\{[\![\lambda_\alpha]\!]\}_{\alpha\in C}$ is generated by each i-th party in the preprocessing phase using a random seed referred to as state_i. For $1 \le i \le n-1$, i-th party also uses state_i to generate the share of $[\![\lambda_{\alpha\cdot\beta}]\!]$, while n-th party uses auxiliaries aux_n as its shares.
– The masked shares $\{\hat{z}_\alpha\}$ of *input wire* of circuit and broadcast messages (denoted by msgs_i) of each of the parties allow for a *deterministic* online phase of the MPC protocol.

C.1 Bare-bones KKW

Next, we describe the bare-bones MPCitH protocol with non-negligible soundness error. To prevent the prover from cheating in the preprocessing phase, the prover must first generate and commit to m executions of the preprocessing stage, and later open all of them except one (corresponding to a verifier's challenge). The unopened material is used for executing the MPC protocol later on. For clarity, we will omit the random coins of the commitment scheme in the description below. We assume that whenever a commitment is opened, the corresponding random coins are provided to the verifier. This results in a 5-move protocol as described below:

1. First, prover runs m independent executions of the preprocessing phase as follows. Prover samples m random values $\{\mathsf{seed}_j\}_{j\in[m]}$. Then each seed_j is used to generate the set of $\{\mathsf{seed}_{j,i}\}_{i\in[n]}$ together with randomness values (we omit these values for simplicity). For each $j \in [m]$, computing $\mathsf{aux}_{j,n}$ as described above, defining $\mathsf{state}_{j,i} := \mathsf{seed}_{j,i}$ for $i \in [n-1]$ and $\mathsf{state}_{j,n} := \mathsf{seed}_{j,n} \| \mathsf{aux}_{j,n}$. Later, prover commits to each state as $com_{j,i} = \mathsf{Com}(\mathsf{state}_{j,i})$, each seed_j as $h_j = H(com_{j,1}, \ldots, com_{j,c})$. Prover then sends $h := H(h_1, \ldots, h_m)$ to verifier.
2. Verifier asks prover to open $m-1$ of the preprocessing material, i.e., all except for the c_1-th instance, where $c_1 \leftarrow\!\!\$\ [m]$.
3. Prover uses the unopened preprocessing material seed_{c_1} to get $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$ and uses them to deterministically simulate the execution of MPC protocol. Finally, prover computes the masked values $\{\hat{z}_\alpha\}$ for the input wires (based on w and $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$) and the broadcast messages from parties during execution $\{\mathsf{msgs}_i\}_{i\in[n]}$.
   Prover sends $\{\mathsf{seed}\}_{j\neq c_1}$ as the opening of all-but-one c-th processing materials and $\{\hat{z}_\alpha\}, h' := H(\mathsf{msgs}_1, \ldots, \mathsf{msgs}_n)$ to verifier.
4. Verifier asks prover to open the views of all parties except for the c_2-th party in the simulation of the MPC protocol, where $c_2 \leftarrow\!\!\$\ [n]$.
5. Prover opens the views by sending $\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}$ along with $com_{c_1,c_2}, \mathsf{msgs}_{c_2}$ that can be used to verify the correctness of the MPC execution.

   Finally, the verifier:

– For $i \neq c_2$, uses $\mathsf{state}_{c_1,i}$ to compute $com_{c_1,i}$ and then combines these with com_{c_1,c_2} to compute $h_{c_1} = H(com_{c_1,1}, \ldots, com_{c_1,n})$.
– For $j \neq c_1$, uses $\{\mathsf{seed}\}_{j\neq c_1}$ to compute $\{h_j\}_{j\neq c_1}$. Then, uses h_{c_1} to check whether $h = H(h_1, \ldots, h_m)$, otherwise outputting reject.
– From $\{\hat{z}_\alpha\}, \{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, \mathsf{msgs}_{c_2}$, simulates the MPC protocol to get $\{\mathsf{msgs}_i\}_{i\neq c_2}$ and the output bit b. If $b = 0$ then it outputs reject.
– Checks whether $h' \neq H(\mathsf{msgs}_1, \ldots, \mathsf{msgs}_n)$ and outputs reject, otherwise accepts.

In Theorem 2.1 of [KKW18], the authors proved that the above protocol has soundness/knowledge error

$$\max\left\{\frac{1}{m}, \frac{1}{n}\right\}$$

, and can be used to construct a ZK argument for NP with negligible soundness error by using parallel repetition. Subsequently, Theorem 7 of [ABB+26] showed that the above protocol has (1) CRSHVZK at both round $i^*_{\mathsf{zk}} = 1$ and at round $i^*_{\mathsf{zk}} = 2$, and (2) is (extended) $(2,2)$-special soundness.

C.2 Cut-and-Choose KKW

The original KKW framework proposed an optimized version of the above protocol using $(m-\tau)$-out-of-m cut-and-choose of the preprocessing phase, leading to a 5-move protocol with soundness/-knowledge error

$$\max_{0\le k\le \tau} \frac{\binom{m-k}{m-\tau}}{\binom{m}{m-\tau}} \cdot \frac{1}{n^{\tau-k}}.$$

In this variant, the prover first generates m independent executions of the preprocessing phase as before, but only opens $m-\tau$ of them (instead of $m-1$) for some $\tau \in [m-1]$. This is specified by the verifier's challenge $\vec{c}_1 = (c_1^1, \ldots, c_1^\tau) \in [m]^\tau$ such that c_1^j's are pairwise distinct, meaning that c_1^j-th execution of the preprocessing phase is left unopened for each $j \in [\tau]$. In Step 3, the τ unopened materials are used for executing τ instances of the online MPC protocol later on. The verifier then asks the prover to open all but one party's view in the simulation of the MPC protocol, where the unopened party is specified by the challenge $\vec{c}_2 \in [n]^\tau$. Detailed construction is described below.

1. First, prover generates and commits m independent executions of the preprocessing phase as shown in step 1 of Section C.1.
2. Verifier asks prover to open $m-\tau$ of the preprocessing material, i.e., all except for the c_1^j-th instances, denote, $\vec{c}_1 = (c_1^1, \ldots, c_1^\tau) \in [m]^\tau$ such that c_1^j's are pairwise distinct.
3. For each $c_1^j \in \vec{c}_1$, $j \in [\tau]$, prover uses the unopened preprocessing material $\mathsf{seed}_{c_1^j}$ to get $\{\mathsf{state}_{c_1^j,i}\}_{i\in[n]}$ and uses them to deterministically simulate the execution of MPC protocol. Finally, prover computes the masked values $\{\hat{z}_\alpha^{c_1^j}\}$ for the input wires (based on w and $\{\mathsf{state}_{c_1^j,i}\}_{i\in[n]}$) and the broadcast messages from parties during execution $\{\mathsf{msgs}_i^{c_1^j}\}_{i\in[n]}$.

   Prover sends $\{\mathsf{seed}_k\}_{k\notin c_1}$ as the opening of $m-\tau$ processing materials and $\forall j \in [\tau]$, $\{\hat{z}_\alpha^{c_1^j}\}$, $h' := H(\mathsf{msgs}_1^{c_1^j}, \ldots, \mathsf{msgs}_n^{c_1^j})$ to verifier.
4. For $j \in [\tau]$, verifier asks prover to open the views of all parties except for the c_2^j-th party in the simulation of the MPC protocol, where $c_2^j \leftarrow\!\$\ [n] \in \vec{c}_2 = (c_2^1, \ldots, c_2^\tau) \in [m]^\tau$.
5. For each $j \in [\tau]$, prover opens the views by sending $\{\mathsf{state}_{c_1^j,i}\}_{i\neq c_2^j}$ along with $com_{c_1^j,c_2^j}, \mathsf{msgs}_{c_2^j}^{c_1^j}$ that can be used to verify the correctness of the MPC execution.

Finally, the verifier checks the consistency of commitments and the correctness of MPC executions.