Critical Rounds in Multi-Round Proofs: Proof of Partial Knowledge, Trapdoor Commitments, and Advanced Signatures

Masayuki Abe[1], David Balbás[2,3], Dung Bui[4], Miyako Ohkubo[5], Zehua Shang[6], Akira Takahashi[7], and Mehdi Tibouchi[1]

[1] NTT Social Informatics Laboratories, Japan
[2] IMDEA Software Institute, Spain
[3] Universidad Politécnica de Madrid, Spain
[4] IRIF, Université Paris Cité, France
[5] NICT, Japan
[6] Kyoto University, Japan
[7] J.P. Morgan AI Research & AlgoCRYPT Center of Excellence, USA

Abstract. Zero-knowledge simulators and witness extractors, initially developed for proving the security of proof systems, turned out to be also useful in constructing advanced protocols from simple three-move interactive proofs. However, in the context of multi-round public-coin protocols, the interfaces of these auxiliary algorithms become more complex, introducing a range of technical challenges that hinder the generalization of these constructions.

We introduce a framework to enhance the usability of zero-knowledge simulators and witness extractors in multi-round argument systems for protocol designs. *Critical-round zero-knowledge* relies on the ability to perform complete zero-knowledge simulations by knowing the challenge of just one specific round in advance. *Critical-round special soundness* aims to address a stringent condition for witness extraction by formalizing it to operate with a smaller tree of transcripts than the one needed for extended extraction, which either outputs the intended witness or solves the underlying hard problem in an argument system. We show that these notions are satisfied by diverse protocols based on MPC-in-the-Head, interactive oracle proofs, and split-and-fold arguments.

We demonstrate the usefulness of the critical round framework by constructing proofs of partial knowledge (Cramer, Damgård, and Schoenmakers, CRYPTO'94) and trapdoor commitments (Damgård, CRYPTO'89) from critical-round multi-round proofs. Furthermore, our results imply advancements in post-quantum secure adaptor signatures and threshold ring signatures based on MPC-in-the-Head, eliminating the need for (costly) generic NP reductions.

Keywords: Multi-Round Proofs, Critical Round, Composition, Proofs of Partial Knowledge, Trapdoor Commitment, MPC-in-the-Head, Adaptor Signatures, Threshold Ring Signatures

Table of Contents

Critical Rounds in Multi-Round Proofs: Proof of Partial Knowledge, Trapdoor Commitments, and Advanced Signatures . . . 1
Masayuki Abe, David Balbás, Dung Bui, Miyako Ohkubo, Zehua Shang, Akira Takahashi, and Mehdi Tibouchi
1 Introduction . . . 3
1.1 Our Contributions . . . 4
2 Technical Overview . . . 6
2.1 The Critical Round Framework . . . 6
2.2 Multi-Round Proof of Partial Knowledge . . . 8
2.3 Trapdoor Commitment and Applications . . . 10
3 Preliminaries . . . 11
3.1 Notation . . . 11
3.2 Public-Coin Proof System . . . 11
3.3 Trapdoor Commitments . . . 12
4 Critical-Round Zero-Knowledge and Soundness . . . 14
4.1 Definitions . . . 14
4.2 Instantiations of Critical Round Proofs . . . 16
5 Multi-Round Proof of Partial Knowledge . . . 19
5.1 Construction . . . 19
5.2 Security . . . 19
5.3 Suppressing Exponential Blow-up via Multi-Round Share-then-Hash . . . 23
6 Trapdoor Commitment from Multi-Round Protocol . . . 24
7 Applications to Advanced Signatures . . . 27
7.1 Adaptor signatures . . . 27
7.2 Threshold ring signatures . . . 27
8 Conclusion . . . 28
A Composition of Multi-Round Protocols via CDS . . . 32
A.1 Monotone Access Structure . . . 32
A.2 Secret Sharing Scheme . . . 33
A.3 Insecurity of Naïve Multi-Round CDS . . . 33
B Composition of Multi-Round Protocols via Share-then-Hash . . . 34
B.1 Extending Predicate Special Soundness . . . 34
B.2 Analysis of Share-then-Hash in the Predicate Special Soundness Framework . . . 36
C Other Definitions . . . 38
C.1 Relations to Other Notions . . . 40
D Critical Rounds in the KKW Framework . . . 41
D.1 KKW Framework . . . 41
D.2 Critical Round in KKW . . . 42
E More Applications . . . 45
E.1 Accumulators . . . 45
E.2 Trapdoor Commitments with Flexible Trapdoor Allocation . . . 48

1 Introduction

Public-coin *three-move proofs* form a foundational building block in cryptographic protocol design. In such a protocol, a prover aims to convince a verifier of a statement x by first sending a commitment a, receiving a random challenge c, and responding with a message z. The canonical notions of security for these protocols are *special soundness (SS)*, which enables witness extraction from two accepting transcripts (a, c, z) and (a, c', z') for $c \neq c'$, and *special honest-verifier zero-knowledge (SHVZK)*, which ensures that a valid-looking transcript can be simulated when the verifier's challenger c is fixed in advance. Three-move protocols are ubiquitous in cryptography, underpinning applications such as identification schemes, signature schemes, and zero-knowledge proofs.

Recent developments have extended this design paradigm to multi-round public-coin protocols, where transcripts are of the form $(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$ and where each challenge c_i is sampled in round i. The classical notion of special soundness [Cra96] generalizes naturally to the $(k_1, \ldots, k_\mu)$-special soundness framework [ACK21,AFR23], in which knowledge extraction is guaranteed given k_i distinct accepting responses to different challenges at each round i, or in other words, given a so-called accepting *tree of transcripts*. For arguments based on computational hardness assumptions, (computational) special soundness [AFR23] is defined with respect to an *extended tree of transcripts*: given such a tree of accepting transcripts, the extended extractor either outputs a witness for the original relation or solves a related hard problem. A corresponding extension of SHVZK also holds in this setting, where *knowledge of the full challenge sequence* $(c_1, \ldots, c_\mu)$ enables simulation of an accepting transcript.

These properties are sufficient to instantiate non-interactive zero-knowledge (NIZK) proofs and signature schemes via the Fiat-Shamir paradigm [FS87], where challenges are deterministically derived from prior transcript data through a random oracle. The multi-round analogue remains secure in the Fiat-Shamir setting: one can rewind the protocol at each round by programming the random oracle (to construct a tree required by special soundness), and simulate all challenges in advance (to rely on SHVZK). However, there are other natural applications where three-move protocols are used but their multi-round analogues either fail to apply or incur fundamental limitations. We examine two examples below.

Proof of Partial Knowledge. Composing proof protocols is a powerful technique for demonstrating partial knowledge of witnesses for compound statements. Consider, for example, a disjunctive statement $A \vee B$, where the prover knows a witness for A but not for B. The prover wishes to convince the verifier that the disjunction holds, without revealing which statement it can actually prove. The Cramer-Shoenmakers-Damgård (CDS) composition [CDS94] achieves this with three-move protocols. The core idea is to treat the challenge c sent from the verifier as a secret to be shared between the two statements. The prover can then answer any challenge c_A for A using its witness, while for B, it can simulate the proof *by running the zero-knowledge simulator* with a fixed challenge share c_B in advance. When the verifier provides the actual challenge c, the prover *completes* the shares by setting $c_A := c \oplus c_B$. This allows the prover to compute the response z_A for challenge c_A using its witness for A, while also using the pre-computed simulation (a_B, z_B) for the fixed challenge c_B. This observation naturally extends to composing statements following any monotone functions efficiently computable with *monotone span programs* (MSP) [CDM00].

One could naïvely extend the CDS approach to multi-round protocols $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ as follows. Standard SHVZK simulation for multi-round protocols typically requires the simulator to know *all* future challenges $(c_1, \ldots, c_\mu)$ in advance. If a prover tries to simulate the proof for B in a multi-round composition, it would need to fix the entire sequence of challenges $(c_1^B, \ldots, c_\mu^B)$ beforehand. The challenges for the A part $(c_1^A, \ldots, c_\mu^A)$ would then be derived from the verifier's actual challenges $(c_1, \ldots, c_\mu)$ and the fixed B-challenges using the share completion mechanism in each round. However, it is well known in the literature that such a straightforward generalization of CDS gives rise to an insecure protocol due to the following "cross attack" on the (knowledge) soundness of the composition [KLP22,FGQ+23,GHAKS23,ABO+24]. Essentially, the naïve multi-round version of CDS allows a cheating prover to fix the challenges for the A-part in some of the rounds and for the B-part in the others, thereby preventing a knowledge extractor from building a full tree of transcripts through rewinding (see also Appendix A.3 for details). While previous works [KLP22,FGQ+23,GHAKS23,ABO+24] have addressed this issue in clever ways, these methods still fall short of matching the original CDS composition: they support only less expressive access

structures and rely on additional assumptions (see Table 1 and Appendix 1.1 for details). As such, composing multi-round protocols while preserving the full potential of the original CDS composition remains an open question.

Trapdoor Commitments. Another simple yet surprising example are trapdoor commitment (TDC) schemes [Dam90,Fis01], which allow a committer to commit to a message m and open it to any different message m' (i.e. equivocate it) only if it knows a trapdoor td. A trapdoor commitment can be constructed from a public-coin three-move protocol Π as follows. The commitment key is an instance x, and the trapdoor is a witness w such that $R(x, w) = 1$ for some hard relation R. To commit to a message m, the committer runs the zero-knowledge simulator of Π using m as a fixed challenge, obtaining a and z. The committer sends $com := a$ as a commitment, which is hiding due to SHVZK of Π, and sends $open = (m, z)$ to open it, which can be verified by running the verification algorithm of Π. Finally, to equivocate a on m' with the knowledge of a witness w, the committer simply runs the honest prover of Π to obtain z' such that (x, a, m', z') is a valid transcript. Binding follows from the special soundness of Π. Given two openings (m, z) and (m', z') such that $m = m'$, one can simply run the extractor on the two transcripts to obtain a witness w such that $R(x, w) = 1$. Note that the extractor can be used as a building block in other protocols as it runs given two valid openings in an offline manner, i.e., without rewinding the adversary. This property is called *offline trapdoor extractability.*

A central question in extending this construction to the multi-round case is: "Where and how should the committed message m be embedded?" If we embed m as the i-th round challenge, the committer still needs to select all the remaining challenges, as otherwise it would not be able to run the zero-knowledge simulator. But in this case, it does not seem possible to rewind the prover to obtain a full tree of transcripts, and let alone achieving the offline witness extraction.

The Challenge. The failures in extending these constructions to the multi-round case seem to stem from two broad issues. First, the standard notion of SHVZK requires the zero-knowledge simulator to fix all challenges in advance, which significantly restricts flexibility in protocol design. This limitation becomes particularly evident when the simulator must be invoked as a component within other protocols, rather than being used solely in the security proof. Second, the requirement to extract a full tree of transcripts that branches at every round to satisfy $(k_1, \ldots, k_\mu)$-SS seems also very rigid, as it hinders rewinding techniques when used in combination with zero-knowledge simulation.

These observations raise the following questions: *Can we identify a more flexible characterization of multi-round public-coin protocols that enables applications beyond the current scope of* $(k_1, \ldots, k_\mu)$*-SS and SHVZK?* And in particular, *can such a framework capture efficient protocols already in use, thereby extending their utility to applications traditionally limited to the three-move setting?*

1.1 Our Contributions

In this paper, we advance the study of multi-round public-coin proofs by introducing *critical rounds*, both from a zero-knowledge and from a special soundness perspective, and by analyzing their implications. We expand on these below and introduce further details, intuition, and (informal) theorem statements in the technical overview in Section 2.

1. *Study of Critical Round Protocols.* We introduce and formalize *critical round zero-knowledge (CRZK)*, where there exists a (critical) round i^*_{zk} such that the transcript can be simulated only by knowing the challenge at round i^*_{zk}. We also introduce *critical round special soundness (CRSS)*, which is met by protocols in which the extractor only needs a sub-tree of the (extended) $\vec{k}$-special soundness tree to retrieve a valid witness. The critical round here is the first branching round i^*_{ss} in the sub-tree of transcripts.
   The relation between CRZK and CRSS is crucial for the security and efficiency of applications, with specific properties requiring a specific relation between i^*_{zk} and i^*_{ss}. Understanding this interplay allows us to design protocols that balance generality and security according to the desired application.
   We then show that the critical-round framework captures important design paradigms, such as those based on MPCitH and IOPs. We prove that the first MPCitH protocol with preprocessing, also known as the KKW protocol [KKW18], satisfies these notions. We also capture

Scheme	Underlying Protocol			Composed Protocol			Composition	Extra Assumptions
	# of Rounds	Soundness	ZK	# of Rounds	Soundness	ZK		
CDS94 [CDS94,CDM00]	3	2-SS	(S)HVZK	3	2-SS	(S)HVZK	MSP	-
FGQRW23 [FGQ+23]	$2\mu+1$	RBRS	HVZK	1	CS	HVZK	1-out-of-n	NPROM, CRS
Speed-Stacking [GHAKS23]	$2\mu+1$	CS	EHVZK	$2\mu+1$	CS	EHVZK	t-out-of-n	NIPBC(CRS)
ABORST24 [ABO+24]	$2\mu+1$	$\vec{k}$-SS	HVZK	$2\mu+1$	$\vec{k}$-SS	SHVCZK	mNC[1]	DualCom
MR-CDS (Sec.5)	$2\mu+1$	$\vec{k}$-SS	CR(S)HVZK	$2\mu+1$	$\vec{k}'$-SS	CR(S)HVZK	MSP(const. n)	-
MR-StH (App.B)	$2\mu+1$	$\vec{k}$-SS	CR(S)HVZK	$2\mu+1$	$(\vec{k}', \boldsymbol{\Phi})$-PSS	CR(S)HVZK	MSP	CRH

Table 1. Multi-round composition methods. Legend: x-(C)SS: (Computational) x-Special Soundness, RBRS: Round-by-Round Soundness, CS: Computational Soundness. (S/E)HV(C)ZK: (Special/Enhanced) Honest Verifier (Computational) ZK. PSS: Predicate Special Soundness. CRH: Collision-Resistant Hash Function.

several popular protocols including PlonK [GWC19], Bulletproofs [BBB+18] and compressed Σ-protocols [AF22].

2. *Multi-Round Proof of Partial Knowledge.* We show that multi-round proofs with the CRZK property are amenable to secure construction of multi-round proofs of partial knowledge via a generalization of the CDS composition, essentially by limiting the application of the CDS technique to the critical round. Compared to existing approaches to multi-round composition, our result supports more expressive access structures than simple disjunctions and thresholds, while introducing no or mild extra assumptions. We give a comparison to other multi-round compositions in Table 1 to show that our composition (Section 5) is as powerful as the original CDS while preserving the zero-knowledge property. While our resulting protocols are relatively simple, the analysis of their knowledge soundness turns out to be surprisingly delicate, due to parameter degradation unique to multi-round protocols. To address this challenge, we provide two flavors of multi-round composition: (1) *Multi-Round CDS*, supporting a constant number of statements with no extra assumptions, and (2) *Multi-Round Share-then-Hash*, enabling the composition of a polynomial number of statements assuming the existence of a collision-resistant hash. The latter can be viewed as the first multi-round generalization of the Share-then-Hash (StH) technique by Abe et al. [AAB+20] that we manage to realize without random oracles, improving on the original scheme.
3. *Trapdoor Commitments from Critical-Round Protocols.* We present a transformation from critical-round proofs to (offline witness extractable) trapdoor commitment schemes which also relies on the properties of critical rounds. These results enable new recipes for the instantiation of the primitives, such as within the MPCitH framework with preprocessing [KKW18]. An advantage of this approach is that one can instantiate universal trapdoor commitments, i.e., where the relation can belong to any NP language $\mathcal{L}$, while using protocols that are efficient in practice. We also construct an *accumulator* that binds multiple messages into a single string.
4. *Further Applications.* As further applications worth mentioning, our result unlocks MPCitH-based instantiations of (1) post-quantum threshold ring signatures and (2) post-quantum adaptor signatures for arbitrary one-way relations. These follow the frameworks from [HS20,LTZ24], that construct these advanced signatures from trapdoor commitments that allow to embed an instance of a particular NP relation to the commitment key. We obtain them solely from (practically efficient) MPCitH, answering the open question posed in both works. Notably, our result circumvents the overhead of generic NP reductions.

Critical Round Zero-Knowledge. There exist multiple notions in the literature that are closely related to our notions of CRZK. One of the closest relations to CRZK is k-zero-knowledge in [GKK+22,DG23], which is a similar concept that is formulated for compiled non-interactive protocols via the Fiat-Shamir transformation. Therefore, it is formalized in the programmable random oracle model. This notion was originally developed to compensate for the absence of rewinding and challenge programmability in simulation-extractable SNARKs [CHM+20,GWC19,Set20,BBB+18]. One can see CRZK as a generalization and refinement of k-zero knowledge for interactive protocols.

Notions of Special Soundness. 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. While our formulation of CRSS departs from $(k_1, \ldots, k_\mu)$-special soundness, it could also start from those variations, with the drawback that their predicate-relative representations are considerably more complex.

Compositions of Proofs. There are many composition methods for three-move proofs in the literature, e.g., [CDS94,CPS+16a,CPS+16b,AAB+20,AAB+21,ACF21,ZLH+22,GGHAK22,ABFV22,ABF+24]. The CDS composition is the most powerful in terms of the expressiveness of the composition and the generality of the underlying protocols, as it enables arbitrary access structures represented by monotone span programs [CDM00]. [AAB+20] extends the underlying protocol to k-special sound ones for $k > 2$ using so-called Share-then-Hash technique, which is emploied with refined security analysis in this paper. Other methods focus on reducing communication complexity or providing additional properties, such as delayed input.

In Table 1, we overview compositions for multi-round protocols. While most of these three-move compositions do not extend to multi-round protocols, Speed-stacking in [GHAKS23] is a multi-round extension of Stacking-Σ [GGHAK22] for three-move compositions. It employs a non-interactive partially binding commitment scheme (NIPBC) that is available in the common reference string (CRS) model making the composed protocol computationally sound. The composition in [FGQ+23] transforms round-by-round sound multi-round protocols into a non-interactive argument system in the non-programmable random oracle model (NPROM) and the CRS model. The composition in [ABO+24] supports compositions in monotone NC^1 (mNC^1) and preserves the soundness of the underlying protocols. It however uses a dual-mode commitment scheme (DualCom) and results in computational zero-knowledge protocols.

Trapdoor commitments and their applications. The idea of constructing a commitment scheme from a three-move public-coin poofs dates back to [Dam90]. There are three-move honest verifier zero-knowledge protocols for NP-complete languages, e.g., [Blu86,GMW91,HV20], that fit the generic construction of TDC, but none of them are practical. On the other hand, there is currently no known method to construct TDCs from more practical multi-round proof systems for NP.

As mentioned before, recent applications of trapdoor commitment include constructions of advanced signature schemes such as threshold ring signatures [HS20] and adaptor signatures [LTZ24]. Simultaneously with our work, [CLTZ24] presents an alternative approach from MPCitH to adaptor signatures. Their scheme incorporates extractable commitments as an additional building block.

2 Technical Overview

We present a technical overview of our contributions. We start by introducing the intuition behind our notions of critical round zero-knowledge and critical round special soundness. Then, we present the main insights of our two primary use-cases for these notions: multi-round protocol composition for proofs of partial knowledge and trapdoor commitments.

2.1 The Critical Round Framework

Critical Round Zero-Knowledge. Traditional formulations of honest-verifier zero-knowledge in the multi-round setting require the simulator to fix all verifier challenges in advance. In our notion of *critical round special honest-verifier zero-knowledge (CRSHVZK, Definition 8)*, simulation is enabled by fixing in advance only a single challenge in a designated *critical round* i^*_{zk}. That is, the simulator receives $c_{i^*_{\mathsf{zk}}}$ as the only challenge fixed in advance, and generates a transcript $(a, c_1, z_1, \ldots, c_{i^*_{\mathsf{zk}}}, \ldots, c_\mu, z_\mu)$

that is indistinguishable from an honestly generated one. Precisely, our notion enforces that the simulator can be decoupled into two (stateful) subroutines: SimA and SimZ. SimA generates the initial message: on input the critical round challenge $c_{i^*_{\mathsf{zk}}}$, it outputs a. SimZ generates the remaining prover messages sequentially: on input an arbitrary challenge c_i, it outputs the i-th round message z_i.

We remark that there exist notions in the literature that are closely related to CRZK, especially k-zero-knowledge in [GKK+22,DG23]. This is a similar concept that is formulated for compiled non-interactive protocols via the Fiat-Shamir transformation (in the programmable random oracle model). One can see CRZK as a generalization and refinement of k-zero knowledge for interactive protocols.

Critical Round Special Soundness. As a generalization of special soundness for Σ-protocols, μ-round multi-round proofs often satisfy $(h_1, \ldots, h_\mu)$-special soundness, requiring the existence of a successful knowledge extractor taking a tree of transcripts as input. However, when the $(h_1, \ldots, h_\mu)$-special soundness is only computational, the guarantee is that the knowledge extractor outputs either a solution to some computationally hard relation R_{hd} *or* a valid witness for NP relation R. As observed in the analysis of real-world protocols, a $(h_1, \ldots, h_\mu)$-tree T is often redundant when it comes to knowledge extraction for R only. That is, only a partial tree of transcripts T^* would be sufficient for extracting a candidate of witness of R. For instance, if the protocol has a typical commit-and-prove structure, a $(k_1, \ldots, k_\mu)$-subtree T^* of $(h_1, \ldots, h_\mu)$-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 binding relation R_{hd}).

We formalize this intuition by formally identifying T^* as a *critical subtree* of *extended tree* T. 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}. In essence, the $(k_1, \ldots, k_\mu)$ critical tree is sufficient to obtain a witness, and the $(h_1, \ldots, h_\mu)$-tree guarantees that the witness is indeed valid. We illustrate this in Figure 1.

We then define our *critical-round special soundness* notion by requiring the existence of a *critical-round extractor*. Following the framework in [AFR23], we define an extended extractor ExExt for extended special soundness (Definition 9) that outputs either a valid witness for relation R or a solution to external hardness assumption R_{hd}. Given a valid extended tree of transcripts T, ExExt succeeds with probability 1. [8] The critical-round extractor CrExt is then defined as a refinement of ExExt that operates on the critical sub-tree T^*. We say that the protocol has CRSS if the following condition is satisfied: as long as successful ExExt does not output a solution to R_{hd} on T as input, CrExt always succeeds in obtaining valid w for R on receiving the corresponding critical subtree T^* of T. As we shall see soon in our trapdoor commitments, this refinement becomes crucial for applications where the protocol explicitly requires a minimal tree of transcripts sufficient for witness extraction.

Instantiations of Critical Rounds in Real-World Protocols. Our definitions are useful insofar as they align with how simulators and extractors are defined for some practical proof systems. In Section 4, we show that many existing proof paradigms including MPCitH and IOP-based proofs can be captured by our framework. Our most relevant instantiation, particularly well-suited for applications, are critical rounds in the MPCitH with preprocessing paradigm [KKW18]. Surprisingly, this protocol even admits different critical-round characterizations as stated below (see Section 4 for details).

Theorem 2 (simplified). *The 5-move interactive honest-verifier zero-knowledge proof in [KKW18] is:*

- *CRSHVZK at round* $i^*_{\mathsf{zk}} = 1$ *or at round* $i^*_{\mathsf{zk}} = 2$.
- $\vec{k}$*-CRSS where* $i^*_{\mathsf{ss}} = 1,\ \vec{k} = (2, 1)$ *or where* $i^*_{\mathsf{ss}} = 2,\ \vec{k} = (1, 2)$.

This implies that the MPCitH protocol provides flexibility in its integration into upper-level protocol designs according to their requirements.

For IOP-based proofs, a prominent example is the Plonk protocol [GWC19], where the verifier ultimately evaluates all the polynomial commitments sent by the prover on a single evaluation point $\mathfrak{z}$, which is the fourth-round challenge in the protocol. As observed in prior work [GKK+22], the knowledge of this challenge in advance suffices to simulate the remainder of the transcript, without

[8] Witness extraction under computational assumptions has been discussed in several works, e.g., [DG23,LPS24,GOP+25]. They typically involve computationally bounded adversaries, while our definition ensures that the extended extractor always succeeds.

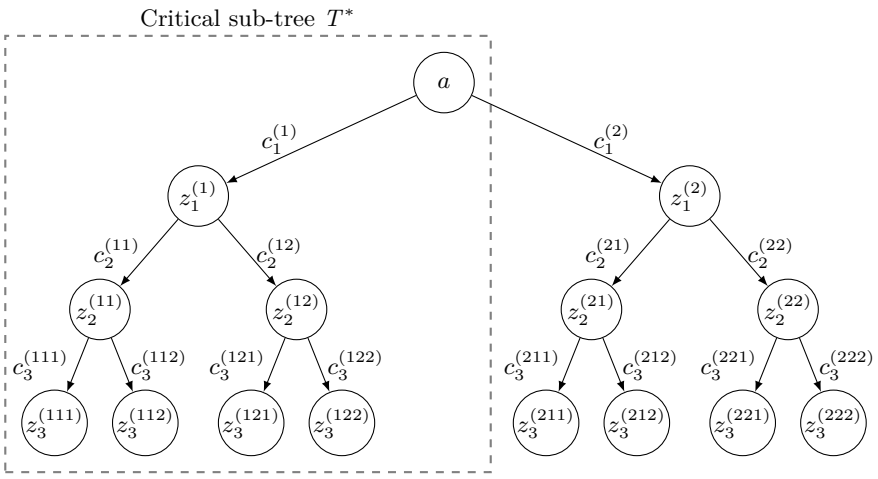

Fig. 1. $(2, 2, 2)$-SS tree T and critical sub-tree T^* for a three-round (seven-move) protocol that is $(1, 2, 2)$-CRSS at round $i^*_{\mathsf{ss}} = 2$, which is the first branch in T^*.

requiring knowledge of other challenges. As we argue in Proposition 1, Plonk admits a CRSHVZK simulator at $i^*_{\mathsf{zk}} = 4$. Besides, we extend the analysis in [LPS24] to argue that Plonk also satisfies CRSS at round $i^*_{\mathsf{ss}} = 4$. Following a similar analysis, we capture Bulletproofs [BBB+18] and compressed Σ-protocols [AF22] in our framework.

2.2 Multi-Round Proof of Partial Knowledge

Composition from Critical Round Protocols. Our work introduces a general framework for securely composing multi-round interactive proofs *in the plain model* for proving partial knowledge. This framework is presented in Section 5, with an extension via the Share-then-Hash technique analyzed in Section B. As recalled in the introduction, a naïve CDS composition quickly runs into soundness problems. Our critical round framework provides a path forward. The crucial insight lies in the properties of CRZK. Recall that a CRZK protocol has a designated critical round i^*_{zk} and a simulator split into $(\mathsf{SimA}, \mathsf{SimZ})$. SimA generates the first message a needing only the critical challenge $c_{i^*_{\mathsf{zk}}}$ in advance. Importantly, the stateful SimZ generates subsequent responses z_i sequentially, taking the actual verifier challenge c_i for round i as input *at that round*. This structure allows us to adapt the CDS strategy effectively:

1. **Non-Critical Rounds ($i \neq i^*_{\mathsf{zk}}$):** For rounds other than the critical one, the prover receives the verifier's challenge c_i. For the statement it holds a witness for (say A), it computes the response z_i^A honestly using the real prover algorithm Z^A. For the statement it is simulating (say B), it uses the stateful simulator SimZ^B with input c_i to compute the response z_i^B. This is possible because SimZ operates round-by-round *after* seeing the challenge.
2. **Critical Round ($i = i^*_{\mathsf{zk}}$):** This round mirrors the logic of the original 3-move CDS. To simulate the B part, the prover must run SimA^B, which requires fixing the critical challenge $c^B_{i^*_{\mathsf{zk}}}$ in advance. The prover samples this value. When the verifier sends the actual challenge $c_{i^*_{\mathsf{zk}}}$ for this round, the prover uses $\mathsf{Complete}$ algorithm of the secret sharing scheme, along with the pre-sampled $c^B_{i^*_{\mathsf{zk}}}$, to determine the necessary challenge $c^A_{i^*_{\mathsf{zk}}}$ for the A part. It then computes $z^A_{i^*_{\mathsf{zk}}}$ honestly using Z^A and the witness for A, and use SimZ^B (with the state from SimA^B) to compute $z^B_{i^*_{\mathsf{zk}}}$ for the fixed challenge $c^B_{i^*_{\mathsf{zk}}}$.

This approach circumvents the soundness issue previously described. The prover is no longer required to fix all challenges for the simulated part upfront. The dynamic nature of SimZ handles the non-critical rounds, while the secret sharing scheme applied *only* at the critical round i^*_{zk} ensures the binding property, analogous to the 3-move case. Therefore, the prover cannot mix-and-match fixed challenges across rounds to defeat the extractor. Our full construction, Π^{cds} (Figure 3), which we call MR-CDS, formalizes this approach for general access structures represented by monotone span programs.

Security of Composed Critical Round Protocols. While MR-CDS is conceptually as simple as the original CDS, we encounter several issues when analyzing the special soundness. In the case of disjunctive statement $A \vee B$, the high-level goal of the analysis is to determine a $(k'_1, \ldots, k'_\mu)$-tree of transcripts T^{cds} for Π^{cds} in such a way that T^{cds} contains *at least* one $(k_1, \ldots, k_\mu)$-tree T, enabling extraction of witness for either A or B. Since secret-sharing of challenge happens in the critical round i^*_{zk}, Π^{cds} can preserve the parameters for rounds $i > i^*_{\mathsf{zk}}$. In the critical round i^*_{zk}, however, distinct challenges (viewed as a reconstructed secret) only guarantee distinct challenges in either of the shares; for $c_{i^*_{\mathsf{zk}}} = c^A_{i^*_{\mathsf{zk}}} \oplus c^B_{i^*_{\mathsf{zk}}}$ and $\hat{c}_{i^*_{\mathsf{zk}}} = \hat{c}^A_{i^*_{\mathsf{zk}}} \oplus \hat{c}^B_{i^*_{\mathsf{zk}}}$, we have that $c^A_{i^*_{\mathsf{zk}}} \neq \hat{c}^A_{i^*_{\mathsf{zk}}}$ or $c^B_{i^*_{\mathsf{zk}}} \neq \hat{c}^B_{i^*_{\mathsf{zk}}}$. While MR-CDS preserves the special soundness parameter $k'_{i^*_{\mathsf{zk}}} = k_{i^*_{\mathsf{zk}}}$ when $k_{i^*_{\mathsf{zk}}} = 2$, it generally grows exponentially with the number of composed protocols n at the critical round i^*_{zk} if $k_{i^*_{\mathsf{zk}}} > 2$. Indeed, the exponential growth of $k'_{i^*_{\mathsf{zk}}}$ limits the practical composition of a polynomial number of protocols. To address this limitation, we generalize the Share-then-Hash method proposed in [AAB+20]. Instead of using the secret shares directly as challenges in the critical round for simulated proofs, our multi-round Share-then-Hash (MR-StH) proof (Π^{sth} in Figure 3), computes the challenges as $c^j_{i^*_{\mathsf{zk}}} \leftarrow H(\ldots, s^j)$ for $j \in [n]$, where $H : \{0,1\}^* \to \mathcal{C}_{i^*_{\mathsf{zk}}}$ is a collision-resistant hash function, each s^j is a share of the i^*_{zk}-th round challenge $s \in S$ sent by the verifier, and the secret domain S is chosen to be larger than the original challenge space $\mathcal{C}_{i^*_{\mathsf{zk}}}$. Soundness analysis for MR-StH again introduces a non-trivial technical challenge. Indeed, the original StH for Σ-protocols [AAB+20] is only known to satisfy somewhat non-standard *statistical special soundness*, where witness extraction succeeds with overwhelming probability given a tree of transcripts produced by an adversary responding to uniformly sampled challenges. However, in the multi-round setting, the relationship between this notion and more standard soundness definitions – such as Fiat-Shamir knowledge soundness – remains unclear. To address this, we conduct the analysis of our multi-round StH within the *predicate special soundness* (PSS) framework recently introduced by Aardal et al. [AAB+24]. This framework enables a rigorous soundness analysis, bringing the dependence of $k'_{i^*_{\mathsf{zk}}}$ down to linear in n, while pushing the exponential term into the numerator of the final knowledge error. The latter can be suppressed by appropriately choosing the size of the share domain S as a denominator. As such, our multi-round StH supports polynomial-size compositions. As [AAB+24] shows PSS implies Fiat-Shamir knowledge soundness in the random oracle model, we can then conclude that MR-StH is suitable for practical applications.

Finally, in both MR-CDS and MR-StH, the parameter k'_i for rounds $i < i^*_{\mathsf{zk}}$ grows linearly with n, reflecting the possibility that the extractor might encounter different qualified sets in an access structure Γ along different branches of the execution tree of transcripts leading up to the critical round. We highlight that showing this linear loss for an arbitrary monotone access structure Γ is also non-trivial. A naïve analysis would require observing every possible qualified set in the access structure k_i times, leading to an exponential loss in k'_i depending on the size of minimal qualified sets.

We state a simplified version of our result below. For details, we refer the readers to Section 5 and Appendix B.

Theorem 3 & 4 (simplified). *Let $\{\Pi^j\}_{j\in[n]}$ be $(2\mu+1)$-move public-coin proof protocols, each being $\vec{k}$-SS and CRZK at the same critical round i^*_{zk}. Let Γ be a monotone access structure. MR-CDS composition $\Pi^{\mathsf{cds}}_\Gamma$ of $\{\Pi^j\}_{j\in[n]}$ for Γ is $\vec{k}'$-SS and CRZK, where*

1. *For round $i > i^*_{\mathsf{zk}}$, $k'_i = k_i$;*
2. *For round $i = i^*_{\mathsf{zk}}$, $k'_i = (k_i - 1)^n + 1$;*
3. *For round $i < i^*_{\mathsf{zk}}$, $k'_i = n(k_i - 1) + 1$.*

Additionally assuming $H : \{0,1\}^* \to \mathcal{C}_{i^*_{\mathsf{zk}}}$ is a collision-resistant hash function, MR-StH composition $\Pi^{\mathsf{sth}}_\Gamma$ of $\{\Pi^j\}_{j\in[n]}$ for Γ is $(\vec{k}', \boldsymbol{\Phi})$-predicate-special-sound where $k'_{i^*_{\mathsf{zk}}} = n(k_{i^*_{\mathsf{zk}}} - 2) + 2$ and k'_i for $i \neq i^*_{\mathsf{zk}}$ are the same as those of $\Pi^{\mathsf{cds}}_\Gamma$.

The result on StH applies to composition of protocols with varying critical rounds, special sound parameters, and challenge spaces. If each composed protocol has critical round i^j_{zk}, then one can set $i^*_{\mathsf{zk}} = \max_j i^j_{\mathsf{zk}}$ and introduce dummy rounds before round i^j_{zk} for each $j \in [n]$; If each protocol has a special soundness parameter k^j_i, then one can set $k_i = \max_j k^j_i$ and consider every protocol $(k_1, \ldots, k_\mu)$-special sound; If each protocol has a challenge space $\mathcal{C}^j_{i^*_{\mathsf{zk}}}$, then one can use CRH $H^j : \{0,1\}^* \to \mathcal{C}^j_{i^*_{\mathsf{zk}}}$ for each $j \in [n]$. Thus, our result allows composition of protocols with diverse structures, as long as each of them has a critical round.

2.3 Trapdoor Commitment and Applications

Our second main application is the construction of trapdoor commitment schemes from critical round multi-round protocols. Recall from the introduction the construction of a trapdoor commitment from a three-move protocol Π. Now, consider an attempt to generalize this construction to the case when Π is a multi-round protocol which is $(h_1, \ldots, h_\mu)$-special sound. The committer must first find a (deterministic) encoding of m into challenges $c_1, \ldots, c_\mu$. Then, it runs the zero-knowledge simulator to get a and $z_1, \ldots, z_\mu$. In the binding game, the adversary will open the commitment a with distinct messages m and m', producing two accepting transcripts $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ and $(a, c'_1, z'_1, \ldots, c'_\mu, z'_\mu)$ that branch at the first round i such that $c_i \neq c'_i$. To reduce such an adversary to $(h_1, \ldots, h_\mu)$-special soundness, we need to build a tree of transcripts that branches in every round. However, this does not seem possible as all challenges are under the control of the adversary. One can try to involve a random oracle in encoding m into $c_1, \ldots, c_\mu$ in a Fiat-Shamir style, but this strategy does not seem to help either. The reason is that the random oracle must be programmed in the committing phase and it results in different commitment a.

Trapdoor Commitment from a Critical Round Protocol. Our solution essentially consists of leveraging the critical round framework to be able to use the Fiat-Shamir transformation in a more flexible way. 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, we can construct our trapdoor commitment scheme 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 $c_i = H_i(x, a, z_1, c_1, \ldots, z_{i-1})$ for $i \neq i^*_{\mathsf{zk}}$, and where z_i is also obtained by running the simulator.
- The commitment is the transcript up to round i^*_{zk} given by $com := (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$.
- The opening is the remaining part of the transcript, given by $open := (z_{i^*_{\mathsf{zk}}}, \ldots, z_\mu)$.

To equivocate the commitment, one simply computes the honest proof for a different message $m' \neq m$.

Hiding and equivocability follow from the zero-knowledge property of Π as in the three-move case. For binding, consider an adversary that outputs $com := (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$ and openings $open, open'$ to two distinct messages m, m'. Thus, the outputs of the adversary can be seen as two full transcripts with the same prefix up to $z_{i^*_{\mathsf{zk}}-1}$, this is,

$$(x, a, c_1, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1}, m, z_{i^*_{\mathsf{zk}}}, \ldots, c_\mu, z_\mu)$$
$$(x, a, c_1, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1}, m', z'_{i^*_{\mathsf{zk}}}, \ldots, c'_\mu, z'_\mu).$$

In the special soundness language, this means that the two transcripts form a subtree with a single branch up to round $i^*_{\mathsf{zk}} - 1$ and two branches at round i^*_{zk}. With this construction, how can we use the adversary to extract a witness for the relation R, leveraging special soundness? The answer is that if Π is CRSS at round $i^*_{\mathsf{ss}} = i^*_{\mathsf{zk}}$ with branching factor $k_{i^*_{\mathsf{ss}}} = 2$, then we can simply rewind the adversary after round i^*_{ss} to obtain the remaining part of the tree of transcripts. To obtain the missing branches, one needs to use a programmable random oracle. Note, however, that programmability is not needed in the first $i^*_{\mathsf{ss}} - 1$ rounds. We picture the construction, as well as the extraction strategy, in Figure 2, where each of the branches correspond to one opening. We also introduce a simplified statement of our main result below. For details, we refer the readers to Section 6.

Theorem 5 (simplified). *Let Π be a $(2\mu+1)$-move public-coin proof protocol for a one-way relation R that is CRZK at round i^*_{zk}, and $k_{i^*_{\mathsf{zk}}} = 2$. Then, our construction is a trapdoor commitment scheme that is hiding and equivocal, and binding in the programmable random oracle model.*

Clearly, not all multi-round protocols that can be captured by the CRSS and CRZK framework satisfy the requirements of Theorem 5.[9] However, as we introduced above, the KKW protocol does satisfy these requirements in any of its two variants. Hence, our construction yields a universal (i.e., for any NP relation) trapdoor commitment scheme from MPCitH which only requires the usage of an MPCitH zero-knowledge simulator, yielding a (potentially practical) post-quantum construction of a trapdoor commitment scheme.

[9] In Section E, we generalize the above construction to work for $k_{i^*_{\mathsf{zk}}} > 2$, although the opening size degrades linearly with $k_{i^*_{\mathsf{zk}}}$.

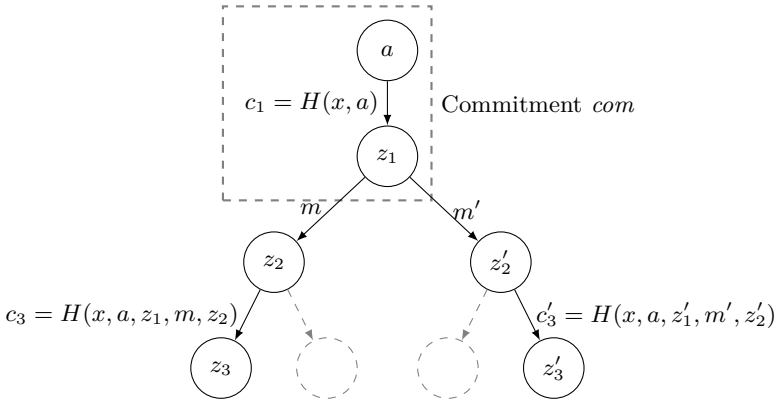

Fig. 2. Visualization of a trapdoor commitment with two openings to m, m' from a three-round protocol that is CRSS and CRZK at round $i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}} = 2$. Dashed arrows and circles can be obtained via rewinding by programming the random oracle H.

Applications. Our universal trapdoor commitment can be used for the construction of threshold ring signatures [HS20] and universal adaptor signatures [LTZ24]. For the latter, the trapdoor commitment must satisfy a stronger *offline trapdoor extractability*, which requires the existence of a straight-line extractor that, given only two transcripts encoding two distinct messages m and m', outputs a witness w. CRSS is essential for this application as it allows for running an extractor on a subtree T^* which, in this case, must be constructed entirely from these two transcripts without rewinding (having a two-tined fork shape). This requirement is satisfied by our trapdoor commitment scheme from MPCitH.

3 Preliminaries

3.1 Notation

Throughout the paper, we use sans-self 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 $[n]$.

3.2 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 choose c_i from $\mathcal{C}_i$ and send 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 ($1 \leq i \leq \mu$) 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.*

We introduce some useful notations related to the tree of transcript. Let T be a $(k_1, \ldots, k_\mu)$-tree of transcripts and $\tau := (a, c_1, z_1, \ldots, c_\mu, z_\mu)$ be a transcript. By $\tau \in T$, we mean that τ is on a path of T. Every node in depth i is indexed by $id := (1, j_1, \ldots, j_{i-1}) \in 1 \times [k_1] \times \cdots \times [k_{i-1}]$. The root node is at depth 1 and indexed by (1). As every node except for the root node has only one incoming edge, every edge is represented by the same index as its destination node. By $\mathsf{path}(T, id)$ we denote the path from the root to node id. A partial transcript assigned on $\mathsf{path}(T, id)$ is denoted by $\mathsf{trans}(T, id)$. By $\mathsf{chal}(T, id)$, we denote a set of challenges assigned to outgoing edges from node id. The partial transcript of τ up to the i-th round is denoted by $\mathsf{prefix}(\tau, i) := (a, c_1, z_1, \ldots, c_i, z_i)$. Below, we present definitions of soundness and zero-knowledge. Additional basic notations can be found in Section C.

Definition 3 (Knowledge Soundness). *An interactive proof system for relation R is knowledge sound with knowledge error ϵ_{ks} if there exists an expected polynomial-time algorithm $\mathsf{Ext}_{\mathsf{ks}}$ called an extractor that, for every algorithm P^*, every $x \in \{0,1\}^\lambda$, and $aux \in \{0,1\}^*$,*

$$\Pr[w \leftarrow \mathsf{Ext}_{\mathsf{ks}}^{P^*(aux)}(x) : R(x, w) = 1] \geq \frac{\Pr[\langle P^*(aux), V\rangle(x) = 1] - \epsilon_{\mathsf{ks}}(\lambda)}{\mathsf{poly}(\lambda)}.$$

It is shown in [ACK21] that $(k_1, \ldots, k_\mu)$-Special Soundness, as defined below, implies knowledge soundness.

Definition 4 ($(k_1, \ldots, k_\mu)$-Special Soundness [ACK21]). *A $(2\mu + 1)$-move public-coin proof protocol is $(k_1, \ldots, k_\mu)$-special sound (SS for short) if there exists a polynomial-time algorithm that, given any accepting $(k_1, \ldots, k_\mu)$-tree of transcripts, outputs w that satisfies $R(x, w) = 1$.*

In the three-move case, the concept of special honest verifier zero-knowledge illustrates that having prior access to the challenge is sufficient to simulate the prover without requiring the witness. This idea extends naturally to the multi-round scenario, where having all challenges in advance enables simulation. Formally:

Definition 5 (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.*

3.3 Trapdoor Commitments

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 6 (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* $\mathcal{A}$,

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

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 also introduce a property called trapdoor extractability, for which one can extract a valid trapdoor from any adversary that breaks binding. It is easy to see that trapdoor extractability implies binding. To define this property, we need to assume the existence of a trapdoor-checking function f_t such that $f_t(ck, td) = 1$ if and only if td is a valid trapdoor for ck. We also define a stronger notion that captures offline extraction, where the extractor $\mathsf{Ext}_{\mathsf{off}}$ has no access to the adversary and thus is required to extract a trapdoor solely from the given openings. On the other hand, the online notion captures a much weaker guarantee requiring the existence of a non-black-box extractor.

Definition 7 (Trapdoor extractability). *A trapdoor commitment scheme is trapdoor extractable if for any PPT adversary* $\mathcal{A}$, *there exists a polynomial-time extractor* Ext *such that*

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

Besides, we say that a trapdoor commitment is offline trapdoor extractable *if there exists an offline polynomial-time extractor* $\mathsf{Ext}_{\mathsf{off}}$ *such that for any PPT adversary* $\mathcal{A}$, *the above probability is negligible.*

4 Critical-Round Zero-Knowledge and Soundness

4.1 Definitions

Recall that the most general form of multi-round zero-knowledge simulator as shown in Definition 5 takes all challenges $(c_1, \ldots, c_\mu)$ as input. Here, we formalize a stronger notion where the entire simulation is possible by knowing only the challenge in the critical round.

Definition 8 (Critical-Round (Special) Honest Verifier Zero-Knowledge). *A $(2\mu+1)$-move public-coin proof protocol is critical-round special honest verifier zero-knowledge (CRSHVZK 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.*

As a weaker variant, if in the above definition SimA only takes x as input, and outputs $(st_1, a, c_{i^*_{\mathsf{zk}}})$, we call it critical-round honest verifier zero-knowledge (CRHVZK for short) at round i^*_{zk}.

In Appendix C.1, we discuss how our CRZK definition is related to other notions such as k-zero knowledge [GKK+22,DG23] and k-special unsoundness [AFK23,BGTZ23].

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. We thus have the following definition for the extended special soundness. Let R_{hd} be a relation for a hardness assumption where for instance y and solution s, it outputs $R_{hd}(y, s) = 1$. It is assumed that given y generated by an instance generator, finding s is hard. A particular example is that y is a hash function chosen uniformly from a family of hash functions and $s = (s_1, s_2)$ is a collision that $y(s_1) = y(s_2)$.

Definition 9 (Extended $\vec{h}$-Special Soundness). *A $(2\mu+1)$-move public-coin argument system for relation R is extended $\vec{h}$-special sound ($\vec{h}$-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 $\vec{h} = (h_1, \ldots, h_\mu)$-tree of accepting transcripts T, outputs either s satisfying $R_{hd}(y, s) = 1$ or w satisfying $R(x, w) = 1$ with probability 1.*

However, in this scenario, the extended tree T would contain more branches than necessary for computing the witness. It may also be the case that the tree building requires programming random oracles which is not feasible in the protocol. Witness extraction is often performed in a real protocol whereas breaking the hardness assumption is not. Therefore, separately handling witness extraction from the extended extractor allows us to identify tighter conditions for the witness extraction to work and to make it executable within the protocol construction. This is analogous to the critical-round zero-knowledge, where the zero-knowledge simulator is designed to be useful in the real protocol run.

Definition 10 (Critical-Round Special Soundness). *A $(2\mu+1)$-move public-coin proof protocol Π for a relation R is $\vec{k}$-critical-round special sound (CRSS for short) at round $i^*_{\mathsf{ss}} \in [\mu+1]$ if the following properties hold:*

- *It is extended $\vec{h}$-special sound for hardness assumption on R_{hd},*
- *there exists a polynomial time algorithm CrExt that, for any extended tree of transcripts T from which ExExt outputs w (but not s) and for every $\vec{k} = (k_1, \ldots, k_\mu)$-subtree T^* of T, outputs w satisfying $R(x, w) = 1$ with probability 1, given x and T^* as input, and*
- *i^*_{ss} is the earliest round where $k_i \geq 2$, i.e., $k_{i^*_{\mathsf{ss}}} \geq 2$ and $k_i = 1$ for all $i < i^*_{\mathsf{ss}}$.*

We show that CRSS implies knowledge soundness. In Theorem 1, we compute the probability that a critical tree T^* with uniformly sampled challenges belongs to an extended tree T from which ExExt outputs w but not s. By definition 10, this means $\mathsf{CrExt}(x, T^*)$ outputs a witness. Furthermore, if R_{hd} is hard to solve and the number of possible critical trees is sufficiently large, then given a critical tree T^* with uniformly sampled challenges, $\mathsf{CrExt}(x, T^*)$ outputs a witness with overwhelming probability.

Theorem 1 (Knowledge Error of Critical-Round Special Soundness). *Let Π be a $(2\mu+1)$-move public coin $\vec{k}$-CRSS protocol (Definition 10). For any statement x, hard instance y, for any adversary $\mathcal{A}$, and for any extended tree T extracted from $\mathcal{A}$ such that $\mathsf{ExExt}(x, T) = (w, s)$, let ϵ be the fraction of trees T such that $R_{hd}(y, s) = 1$, i.e., such that extended extraction solves R_{hd}. Then, for any statement x, first transcript message a, and for any PPT adversary $\mathcal{A}$ with non-negligible success probability δ, the following holds:*

$$\Pr\left[\begin{array}{l} T^* \text{ is a } (k_1, \ldots, k_\mu)\text{-tree} \\ \text{of accepting transcripts} \\ \wedge \left(\forall i \in [\mu-1], \forall id \in \overrightarrow{[k_i]}, \right. \\ \left. \quad \mathsf{chal}(T^*, id) = \{c_{id||j}\}_{j\in[k_{i+1}]}\right) \\ \wedge\ \mathsf{root}(T^*) = a \\ \wedge\ R(x, w) = 0 \end{array} \middle| \begin{array}{l} \forall i \in [\mu], \forall id \in \overrightarrow{[k_i]}, c_{id} \leftarrow\$\ \mathcal{C}_i \\ T^* \leftarrow \mathcal{A}(x, a, \{c_{id}\}_{id\in\overrightarrow{[k_\mu]}}) \\ w \leftarrow \mathsf{CrExt}(x, T^*) \end{array}\right] \leq \epsilon_{\mathsf{crss}},$$

where $\overrightarrow{[k_i]} = 1 \times [k_1] \times \cdots \times [k_i]$ and $\epsilon_{\mathsf{crss}} = \frac{\prod_{i\in[\mu]}(h_i - k_i + 1)^{\prod_{j=1}^{i-1} k_j}}{\delta^{\prod_i k_i} \cdot \prod_i \binom{|\mathcal{C}_i|}{k_i}^{\prod_{j=1}^{i-1} k_j}} + \epsilon \cdot \prod_{i\in[\mu]} \binom{h_i}{k_i}^{\prod_{j=1}^{i-1} k_j}$. In particular, if both ϵ and $\left(\delta^{\prod_i k_i} \cdot \prod_i \binom{|\mathcal{C}_i|}{k_i}^{\prod_{j=1}^{i-1} k_j}\right)^{-1}$ are negligible, then CrExt succeeds with overwhelming probability.

Proof. Without loss of generality, fix the statement x, hard instance y and first message a. Given a critical $(k_1, \ldots, k_\mu)$-tree of accepting transcripts T^*, it must match one of the following cases:

- Case 1: There exists an extended tree T such that $T^* \subset T \bigwedge \mathsf{ExExt}(x, T) = (w, s) \bigwedge R(x, w) = 1 \bigwedge R_{hd}(y, s) = 0$. This is, the extended extractor outputs a valid witness for R and not a solution for R_{hd}. In this case, $\mathsf{CrExt}(x, T^*) = w \bigwedge R(x, w) = 1$ with probability 1 due to Definition 10, which specifies that CrExt extracts w if ExExt outputs w (and not s for R_{hd}).
- Case 2: There exists an extended tree T such that $T^* \subset T \bigwedge \mathsf{ExExt}(x, T) = (w, s) \bigwedge R_{hd}(y, s) = 1$. This is, the extended extractor outputs a solution to the hard relation R_{hd} (it might also be that $R(x, w) = 1$, but the primary concern here is the solution to R_{hd}).
- Case 3: For any extended tree T such that $\mathsf{ExExt}(x, T) = (w, s) \bigwedge (R(x, w) = 1 \vee R_{hd}(y, s) = 1)$, it holds that $T^* \not\subset T$. In this scenario, T^* is not a subtree of any extended tree from which ExExt could successfully extract a witness for R or a solution for R_{hd}.

We compute the probability that given a T^* with challenges sampled uniformly at random such that $\mathsf{CrExt}(x, T^*)$ is supposed to output a valid witness w, then T^* belongs to Case 1, denoted by Pr[Case 1].

We denote by M the total number of all accepting critical $(k_1, \ldots, k_\mu)$-trees T^*, and by N the total number of all accepting extended $(h_1, \ldots, h_\mu)$-trees T. The success probability δ of an adversary in constructing accepting transcripts affects the number of such trees. Specifically, if an adversary can only successfully complete a fraction δ of transcripts, the number of fully accepting trees is scaled with a δ^K factor, where K denotes the number of leaf nodes. For a $(k_1, \ldots, k_\mu)$-tree, $K = \prod_{i\in[\mu]} k_i$. Given that the number of all possible trees with challenges sampled uniformly at random from challenge space is $\prod_i \binom{|\mathcal{C}_i|}{k_i}^{\prod_{j=1}^{i-1} k_j}$, we have $M = \delta^{\prod_i k_i} \cdot \prod_i \binom{|\mathcal{C}_i|}{k_i}^{\prod_{j=1}^{i-1} k_j}$. Similarly, $N = \delta^{\prod_i h_i} \cdot \prod_i \binom{|\mathcal{C}_i|}{h_i}^{\prod_{j=1}^{i-1} h_j}$.

We first compute the fraction of critical trees T^* in Case 3 among all possible T^*s. A T^* is in Case 3 if it cannot be part of any T from which ExExt extracts a witness or a hard solution. The number of such "isolated" T^*s must be small, since a full extended tree T can be composed by enough number of critical trees T^*. Let M_3 be the maximum number of T^*s that are in Case 3. We have Pr[Case 3] $= \frac{M_3}{M}$ and

$$M_3 < \prod_{i\in[\mu]} (h_i - k_i + 1)^{\prod_{j=1}^{i-1} k_j}.$$

Next, we compute the fraction of T^*s as parts of an extended tree T leading to extracting a solution for R_{hd} (Case 2) among all T^*s not in Case 3. Assume that a fraction ϵ of all possible extended trees T will have the extractor ExExt output a solution s for $R_{hd}(y,s)=1$. Let N_H be the number of such Ts. We have $N_H = \epsilon N$.

Let M_2 be the number of T^*s such that $T^* \subset T$ for some T where $\mathsf{ExExt}(x,T) = (w,s) \wedge R_{hd}(y,s)=1$. The number of distinct T^*s that can be formed as subtrees of a single T is at most $\prod_{i\in[\mu]} \binom{h_i}{k_i}^{\prod_{j=1}^{i-1} k_j}$. Let this factor be $K_S = \prod_{i\in[\mu]} \binom{h_i}{k_i}^{\prod_{j=1}^{i-1} k_j}$. Hence, an upper bound on the number of T^*s that could fall into Case 2, denoted by M_2, is:

$$M_2 \leq N_H \cdot K_S = \epsilon N \cdot K_S.$$

Let $M_{\neg 3}$ be the number of T^*s that are not in Case 3. These are the T^*s that are subtrees of at least one T where ExExt successfully extracts either a witness for R or a solution for R_{hd}. We argue that $M_{\neg 3} > N$. Notice that two different extended trees T will have at least one different critical sub-tree T^*. This implies that the collection of all T^*s that are parts of *any* of these N extended trees must number at least N. Thus, $M_{\neg 3}$ satisfies $M_{\neg 3} > N$.

The probability that a T^* (which is not in Case 3) exists in an extended tree leading to extracting a solution for R_{hd} is Pr[Case 2 | Not Case 3]. We have

$$\Pr[\text{Case 2} \mid \text{Not Case 3}] \leq \frac{M_2}{M_{\neg 3}} < \frac{\epsilon N \cdot K_S}{N} = \epsilon \cdot K_S = \epsilon \cdot \prod_{i\in[\mu]} \binom{h_i}{k_i}^{\prod_{j=1}^{i-1} k_j}.$$

The probability that a uniformly sampled T^* results in $\mathsf{CrExt}(x, T^*)$ outputting a valid witness w for R (and not a solution for R_{hd}) is Pr[Case 1]. This occurs if T^* is not in Case 3, and, given it's not in Case 3, it does not fall into Case 2.

$$\begin{aligned}
\Pr[\text{Case 1}] &= \Pr[\text{Not Case 2} \mid \text{Not Case 3}] \cdot \Pr[\text{Not Case 3}] \\
&= (1 - \Pr[\text{Case 2} \mid \text{Not Case 3}]) \cdot (1 - \Pr[\text{Case 3}]) \\
&\geq \left(1 - \frac{M_2}{M_{\neg 3}}\right) \cdot \left(1 - \frac{M_3}{M}\right).
\end{aligned}$$

Thus, the probability that $\mathsf{CrExt}(x, T^*)$ yields a witness w with $R(x,w)=1$ (implying it's Case 1) is:

$$\begin{aligned}
\Pr_{T^* \leftarrow\$ \{T^*\}} \begin{bmatrix} w \leftarrow \mathsf{CrExt}(x, T^*) \\ \wedge R(x,w) = 1 \end{bmatrix} &\geq \left(1 - \frac{M_3}{M}\right)\left(1 - \frac{M_2}{M_{\neg 3}}\right) \\
&> 1 - \frac{M_3}{M} - \frac{M_2}{M_{\neg 3}} \\
&> 1 - \frac{\prod_{i\in[\mu]} (h_i - k_i + 1)^{\prod_{j=1}^{i-1} k_j}}{\delta^{\prod_i k_i} \cdot \prod_i \binom{|\mathcal{C}_i|}{k_i}^{\prod_{j=1}^{i-1} k_j}} - \epsilon \cdot \prod_{i\in[\mu]} \binom{h_i}{k_i}^{\prod_{j=1}^{i-1} k_j}.
\end{aligned}$$

Finally, if ϵ and $1/M = 1/(\delta^{\prod_i k_i} \cdot \prod_i \binom{|\mathcal{C}_i|}{k_i}^{\prod_{j=1}^{i-1} k_j})$ is negligible, then the above event happens with overwhelming probability. □

In Appendix C.1, we discuss how our CRZK definition is related to other notions such as k-zero knowledge [GKK+22,DG23] and k-special unsoundness [AFK23,BGTZ23]. We also discuss possible relaxations of CRSS.

4.2 Instantiations of Critical Round Proofs

MPC-in-the-Head based Proofs. We cast the honest-verifier zero-knowledge proof in [KKW18] (denoted by KKW) and formally prove that their 5-move protocol satisfies CRZK and CRSS. We note that many other multi-round protocols that follow the MPCitH with preprocessing paradigm also fall into this class, including (but not limited to) [KKW18,BN20,KZ22,FJR22].

In the KKW framework, the prover runs an MPC protocol that evaluates a boolean circuit C on an input w (witness), commits to the views of all parties, and then opens all-but-one of these views to

the verifier. This MPC protocol 1) is secure against semi-honest all-but-one corruptions, hence there exists a simulator Sim_p that outputs simulated consistent views of $n-1$ parties, and 2) is executed in a *deterministic* manner by using a preprocessing material sampled *independently* of the witness w. To prevent the prover from cheating in the preprocessing phase, the prover follows the cut-and-choose paradigm, i.e., first generates and commits to m executions of the preprocessing stage, and later opens all of them except one (corresponding to a verifier's challenge). The unopened material is used for executing the MPC protocol later on.[10]

In Theorem 2, we show that KKW is flexible and can be analyzed as (1) CRZK at either $i^*_{\mathsf{zk}} = 1$ or $i^*_{\mathsf{zk}} = 2$, and (2) CRSS also at either $i^*_{\mathsf{ss}} = 1$ or $i^*_{\mathsf{ss}} = 2$. The formal proof is shown in Theorem 8 together with the detailed description of the KKW framework in Section D.

Theorem 2. *Given the 5-move interactive honest-verifier zero-knowledge proof in [KKW18] (denoted as KKW), assuming that the hash function used is collision-resistant and the commitment scheme used is computationally binding and hiding, then KKW is flexible and can be:*

- *Critical-round special honest-verifier zero-knowledge at round $i^*_{\mathsf{zk}} = 1$ or at round $i^*_{\mathsf{zk}} = 2$.*
- *$\vec{k}$-critical-round special sound where $i^*_{\mathsf{ss}} = 1$, $\vec{k} = (2, 1)$ or where $i^*_{\mathsf{ss}} = 2$, $\vec{k} = (1, 2)$.*

Let (a, c_1, z_1, c_2, z_2) be a transcript of the KKW protocol. We provide the intuition behind the formal proof (in Theorem 8) of the flexibility property of KKW with respect to CRSS and CRZK.

CRSS. We prove that KKW satisfies CRSS for both $i^*_{\mathsf{ss}} = 1$ and $i^*_{\mathsf{ss}} = 2$ by first constructing an extended extractor ExExt for each of the cases to show that KKW is extended $(2, 2)$-special sound. From ExExt, we then construct an extractor CrExt that, given access to an accepting prover, extracts a valid witness (and not a solution of R_{hd} breaking commitment binding or hash function collision resistance) from a subtree of transcripts, while ensuring that this is defined according to the CRSS definition. Given an accepting $(2, 2)$-tree of transcripts T,

- We construct a different extractor ExExt for each case, requiring only two transcripts:
  - $(a, c_1, z_1, *, *)$ and $(a, c'_1, z'_1, *, *)$ in the first case, where $i^*_{\mathsf{ss}} = 1$,
  - or (a, c_1, z_1, c_2, z_2) and $(a, c_1, z_1, c'_2, z'_2)$ in the second case, where $i^*_{\mathsf{ss}} = 2$.

  The extractor ExExt successfully extracts a unique value w with probability 1. Moreover, this extracted w is indeed a valid witness. Otherwise, using another accepted transcript, ExExt relies on the following properties:
  - the parties' views (broadcast messages) are committed in the third move via a hash function, and
  - all states are committed in the first move using a hash function and a commitment scheme,

  to either output a collision in the hash function H or two distinct inputs that break the binding property of the commitment scheme.
- Therefore, for both cases, given a deterministic tree T, the extractor CrExt is constructed in the same way as ExExt, stopping as soon as it obtains w. Since ExExt succeeds on extracting a valid witness from T, it follows that w is a valid witness.

CRZK. We prove that KKW is CRZK at either $i^*_{\mathsf{zk}} = 1$ or $i^*_{\mathsf{zk}} = 2$ by constructing two simulators $(\mathsf{SimA}, \mathsf{SimZ})$ that simulate the real execution.

- For $i^*_{\mathsf{zk}} = 1$, $\mathsf{SimA}(x, c_1) \to (st_1, a)$ takes the first-round challenge c_1 as input. For $i^*_{\mathsf{zk}} = 2$, $\mathsf{SimA}(x, c_2) \to (st_1, a)$ instead takes the second-round challenge c_2 as input. In both cases, SimZ uses SimA's output along with c_1 and c_2 to complete the simulation of the real transcripts.
- The construction of SimA also differs in each case. For $i^*_{\mathsf{zk}} = 1$, since SimA knows the challenge c_1, it simulates the c_1-th preprocessing step by emulating the auxiliary values aux to ensure the correctness of the MPC output. On the other hand, for $i^*_{\mathsf{zk}} = 2$, SimA knows the index of all-but-one party in the MPC protocol, allowing it to use the MPC simulator to simulate the views of the remaining party.

[10] This cut-and-choose strategy receives several optimizations, e.g. [BN20]. Their zero-knowledge simulation strategy exploited in our analysis remains unchanged.

IOP-based Proofs. For SNARKs based on IOPs [RRR16,BCS16] and polynomial IOPs [BFS20], the proof of knowledge soundness does not involve building a tree of transcripts as in the KKW framework. Rather, it often relies on the extractability of a polynomial commitment scheme, which is usually proven in the algebraic group model or relies on knowledge assumptions where no rewinding of the adversary is required. In these cases, the knowledge extractor can be naturally adjusted to the algebraic group model (AGM) or to extra assumptions such that it is given additional auxiliary information (e.g., the algebraic representations of all group elements) extracted from the adversary. The shape of the extended tree of transcripts T for extended $\vec{h}$-special soundness is related to the polynomial evaluation checks and batching operations which usually rely on the statistical soundness of the Schwartz-Zippel lemma. These protocols can be regarded as $(1,\ldots,1)$-CRSS with critical round $i^*_{\mathsf{ss}} = \mu+1$. Jumping ahead, $i^*_{\mathsf{ss}} = \mu+1$ requires some care within our applications as it causes $i^*_{\mathsf{ss}} \neq i^*_{\mathsf{zk}}$. To avoid reliance on the AGM, in some cases, one can analyze the interactive proof systems underlying these SNARKs in the standard model.

Plonk. Plonk [GWC19] is a SNARK that is built by compiling a multi-round polynomial IOP via the Fiat-Shamir transform, where the polynomial oracles are instantiated with the KZG polynomial commitment scheme [KZG10]. At a high level, the prover first commits to several polynomials encoding the witness values, and then proves circuit satisfiability via multiple polynomial checks. In the standard model, a recent work by Lipmaa, Parisella, and Siim [LPS24] characterizes Plonk as a computationally (i.e., extended) $\vec{h}$-special-sound five-round protocol. [11] Concretely, they obtain that for an upper bound of n circuit gates, Plonk is extended $\vec{h}$-special-sound for $\vec{h} = (3n+1, 3n+1, 3, 4n+21, 6)$. We extend their analysis by noticing that their extractor is guaranteed to work (either by extracting a valid witness or by breaking the binding of the polynomial commitment scheme) given a $(1,1,1,4n+21,6)$ subtree of transcripts T^* of the extended special sound tree of transcripts T. Therefore, five-round Plonk satisfies CRSS where $i^*_{\mathsf{ss}} = 4$. A drawback is that k_4 grows linearly with the witness size, which reduces the efficiency of our applications.

For CRZK, the main observation is that the first three responses of the Plonk prover consist of commitments to various polynomials, (primarily as part of batching techniques) whereas the fourth and the fifth prover responses contain only evaluations and opening proofs. All the polynomial evaluations used by the verifier are carried out at the same evaluation point, $\mathfrak{z}$, which is the fourth challenge sent by the verifier. Hence, it is possible to simulate an entire transcript by knowing only $\mathfrak{z}$ in advance. This observation, albeit with the different purpose of showing simulation extractability, was done in [GKK+22]. One caveat is that, to enable this simulation strategy without any simulation trapdoor, the KZG polynomial commitments need to be randomized as pointed out by [KPT23] – we refer to their work for details. Overall, we conclude:

Proposition 1. *The 5-round Plonk protocol described in [LPS24] satisfies $\vec{k}$-CRSS for $i^*_{\mathsf{ss}} = 4$ in the standard model, where $\vec{k} = (1,1,1,4n+21,6)$. In the AGM, Plonk is $\vec{k}$-CRSS for a final round $i^*_{\mathsf{ss}} = 6$, where $\vec{k} = \vec{1}$. Moreover, Plonk satisfies critical-round zero knowledge at round $i^*_{\mathsf{zk}} = 4$.*

Bulletproofs and Compressed Σ-protocols. Another prominent example of a multi-round proof is Bulletproofs [BCC+16,BBB+18], a widely used transparent proof system secure under the discrete logarithm assumption. In Bulletproofs, the prover commits to a length d witness and then reduces the proving statement to another one of length $d/2$. After $\log d$ rounds, the prover opens the commitments and the verifier can check the satisfiability of the relation.

For CRZK, [DG23] proves the 3-ZK notion [GKK+22] of the protocols[12], As their simulator works by obtaining the third round challenge x as input and then the following rounds can be simulated sequentially, their analysis directly implies CRZK at round $i^*_{\mathsf{zk}} = 3$. As such, the following statements hold:

Proposition 2. *Bulletproofs for aggregate range proof (BP-ARP) [BBB+18] is a $(2\mu+1)$-move protocol where $\mu = 4 + \lceil \log(\ell d) \rceil$, ℓ is the number of aggregated commitments, and d is the maximum bit*

[11] We note that Plonk is usually regarded as a four-round protocol, but the first round is divided into two in their analysis.

[12] In [DG23], they call it 2-ZK grouping the first two challenges y, z together as a single round. In this work, we view them as challenges from two separate rounds so that the number of rounds matches the dimension of the special soundness parameter $\vec{k}$.

length of the committed values. It is $\vec{k}$- special sound, where $\vec{k} = (\ell d, \ell + 2, 3, 2, 7, \ldots, 7)$. Moreover, BP-ARP is CRZK at round $i^*_{\mathsf{zk}} = 3$ from [DG23, Lemma 6.4].

Proposition 3. *Bulletproofs for arithmetic circuit satisfiability proof (BP-ACS) is a $(2\mu + 1)$-move protocol where $\mu = 4 + \lceil \log(\ell) \rceil$ and ℓ is the number of multiplication gates in the circuit. It is $\vec{k}$-special sound, where $\vec{k} = (\ell, Q + 1, 7, 2, 7, \ldots, 7)$, and Q is the number of linear constraints. Moreover, BP-ACS is CRZK at round $i^*_{\mathsf{zk}} = 3$ from [DG23, Lemma C.2].*

A close variant of Bulletproofs is compressed Σ-protocols [AF22]. As their special HVZK simulator only requires knowing the first round challenge in advance and then it can produce a transcript honestly while receiving the following challenges, we observe that $i^*_{\mathsf{zk}} = 1$. We thus get the following statement:

Proposition 4. *The compressed Σ-protocol for discrete logarithm relation [AF22] is a $(2\mu+1)$-move protocol where $\mu = \lceil \log(n + 1) \rceil + 1$ and n is the number of group generators. It is $\vec{k}$-special sound, where $\vec{k} = (2, 2, 3, \ldots, 3)$. Moreover, it is CRZK at round $i^*_{\mathsf{zk}} = 1$ from [AF22, Theorem 3].*

5 Multi-Round Proof of Partial Knowledge

5.1 Construction

In this section, our goal is to build a proof system for a logically composed relation using proof systems for atomic relations in a black-box manner. We assume the reader is familiar with the concept of monotone access structures (Section A.1) and semi-smooth secret sharing schemes (Section A.2). These relations can be logically composed using a monotone access structure Γ. That is, given n public-coin protocols Π^j for relation R^j, we define a composed protocol $\Pi^{\mathsf{cds}}_\Gamma$ for relation $R_\Gamma((x^1, \ldots, x^n), (w^1, \ldots, w^n))$ that outputs 1 if and only if $\exists A \in \Gamma, \forall j \in A,\ R^j(x^j, w^j) = 1$. Thus, the prover will know valid witnesses for some of the relations (generating qualified transcripts) and will need to simulate the remaining non-qualified transcripts. In brief, given multi-round proofs that are special sound and CRSHVZK at critical round $i^*_{\mathsf{zk}} = i^*$, the prover in our composition follows the 3-move CDS composition framework as below:

- For steps $2i+1$ ($i \neq i^*$), the prover simulates the non-qualified transcripts and honestly generates the qualified ones based on the challenge sent by the verifier at step $2i$.
- For step $2i^* + 1$ at the critical round, the prover samples the critical-round challenges of non-qualified transcripts in advance. After receiving the challenge from the verifier, the prover completes the remaining challenges for the qualified transcripts via the share completion algorithm of SSS_{Γ^*}.

Our MR-CDS composition, $\Pi^{\mathsf{cds}}_\Gamma$, is presented in Fig. 3. The private input to the prover algorithm is a set of witnesses $\{w^j\}_{j \in A}$ that $A \in \Gamma$. The ingredients of our protocol include a semi-smooth secret sharing scheme SSS_{Γ^*} over secret domain S and n CRSHVZK protocols $\{\Pi^j\}_{j \in [n]}$ at critical round i^*. Note that efficient SSS_{Γ^*} exists for any Γ^* (and thus Γ) that is recognized by monotone span program [CDM00]. For simplicity, we assume that the share domain of SSS_{Γ^*} matches the challenge space $\mathcal{C}_{i^*}$ for $\Pi^{\mathsf{cds}}_\Gamma$, while this restriction becomes superfluous for our MR-StH composition, $\Pi^{\mathsf{sth}}_\Gamma$, presented in Section 5.3. Following [CDS94], if challenges are longer than shares, one can simply take the first appropriate number of bits of the challenge to be the corresponding share. The completion process can be also postprocessed by simply copying the bits of the shares and padding with random bits to define the corresponding challenges.

5.2 Security

To illustrate the intuition, we first informally discuss the security of $\Pi^{\mathsf{cds}}_\Gamma$ when $i^* = i^*_{\mathsf{zk}} = 1$ and $k_{i^*} = 2$. Completeness of $\Pi^{\mathsf{cds}}_\Gamma$ follows by inspection: since Π^j for $j \in [n]$ are complete and CRSHVZK, V^j always accepts. Moreover, the share completion property of SSS_{Γ^*} guarantees $\mathsf{CheckShares}_{\Gamma^*}$ always outputs 1. CRSHVZK of $\Pi^{\mathsf{cds}}_\Gamma$ is also straightforward. One can construct a simulator, which, instead of running an honest prover algorithm, executes CRHVZK simulators for all $j \in [n]$ on input

Composition Compiler $\Pi_{\Gamma}^{\mathsf{sth}}$ and $\Pi_{\Gamma}^{\mathsf{cds}}$.

Statement: $\vec{x} = \{x^j\}_{j\in[n]}$.

Witness: $\{w^j\}_{j\in A}$ such that $A \in \Gamma$ and $R^j(x^j, w^j) = 1, \forall j \in A$.

Step 1 : Prover computes as follows:

- Sample $\{s^j\}_{j\in\bar{A}} \leftarrow D_{\bar{A}}$.
- **for all** $j \in \bar{A}$: Compute $c_{i^*}^j = H(\Gamma, \vec{x}, j, s^j)$; Set $c_{i^*}^j = s^j$;
  Simulate $(st_1^j, a^j) \leftarrow \mathsf{SimA}^j(x^j, c_{i^*}^j)$.
- **for all** $j \in A$: Compute $(st_1^j, a^j) \leftarrow \mathsf{A}^j(x^j, w^j; r^j)$.
- Send $\{a^j\}_{j\in[n]}$ to the verifier.

Repeat the following for $i = 1, \ldots, \mu$:

Step $2i$ $(i \neq i^*)$: Verifier samples $c_i \leftarrow\!\!\$\ \mathcal{C}_i$ and sends c_i to the prover.

Step $2i^*$: Verifier samples $s \leftarrow\!\!\$\ S$ and sends s to the prover.

Step $2i+1$ $(i \neq i^*)$: Prover computes as follows:

- **for all** $j \in \bar{A}$: Compute $(st_{i+1}^j, z_i^j) \leftarrow \mathsf{SimZ}^j(st_i^j, c_i)$.
- **for all** $j \in A$: Compute $(st_{i+1}^j, z_i^j) \leftarrow \mathsf{Z}^j(st_i^j, c_i)$.
- Send $\{z_i^j\}_{j\in[n]}$ to the verifier.

Step $2i^*+1$: Prover computes as follows:

- Compute $\{s^j\}_{j\in A} \leftarrow \mathsf{Complete}_{\Gamma^*}(s, \{s^j\}_{j\in\bar{A}})$.
- **for all** $j \in \bar{A}$: Compute $(st_{i^*+1}^j, z_{i^*}^j) \leftarrow \mathsf{SimZ}^j(st_{i^*}^j, c_{i^*}^j)$.
- **for all** $j \in A$: Compute $c_{i^*}^j = H(\Gamma, \vec{x}, j, s^j)$; Set $c_{i^*}^j = s^j$;
  Compute $(st_{i^*+1}^j, z_{i^*}^j) \leftarrow \mathsf{Z}^j(st_{i^*}^j, c_{i^*}^j)$.
- Send $\{z_{i^*}^j, s^j\}_{j\in[n]}$ to the verifier.

Verification : Verifier computes as follows:

- **for all** $j \in [n]$: Compute $c_{i^*}^j = H(\Gamma, \vec{x}, j, s^j)$; Set $c_{i^*}^j = s^j$;
- **for all** $j \in [n]$: **assert** $\mathsf{V}^j(x^j, \pi^j = (a_1^j, c_1^j, z_1^j, \ldots, c_\mu^j, z_\mu^j)) = 1$,
  where $c_i^j := c_i$ for $i \neq i^*$;
- **assert** $\mathsf{CheckShares}_{\Gamma^*}(s, \{s^j\}_{j\in[n]}) = 1$.

Fig. 3. Our multi-round composition $\Pi_{\Gamma}^{\mathsf{cds}}$ (MR-CDS) and $\Pi_{\Gamma}^{\mathsf{sth}}$ (MR-StH) with critical round $i^* = i_{\mathsf{zk}}^*$. The code highlighted in solid box is only executed by $\Pi_{\Gamma}^{\mathsf{cds}}$, while the code in dashed box is only executed by $\Pi_{\Gamma}^{\mathsf{sth}}$, respectively. $\Pi_{\Gamma}^{\mathsf{cds}}$ assumes $\mathcal{C}_{i^*}$ to be identical to the share space of the secret sharing scheme SSS_{Γ^*}. $\Pi_{\Gamma}^{\mathsf{sth}}$ relies on a collision-resistant hash $H : \{0,1\}^* \to \mathcal{C}_{i^*}$.

$\{c_{i^*}^j\}_{j\in[n]} = \{s^j\}_{j\in[n]} \leftarrow \mathsf{Share}_{\Gamma^*}(s)$. The perfect hiding and share completion properties of SSS_{Γ^*} guarantee that $\{s^j\}_{j\in[n]}$ is identically distributed in real and simulated transcripts.

To prove $\vec{k}'$-SS of $\Pi_{\Gamma}^{\mathsf{cds}}$ from $\vec{k}$-SS of Π^j, we need to carefully set the parameters $\vec{k}'$. Since any $\vec{k}'$-tree of transcripts for $\Pi_{\Gamma}^{\mathsf{cds}}$ can be decomposed into n individual trees T^j for Π^j, we can conclude the existence of a valid extractor for $\Pi_{\Gamma}^{\mathsf{cds}}$ by showing the existence of some $A \in \Gamma$ such that for all $j \in A$, T^j contains a $\vec{k}$-tree of accepting transcripts. We split the analysis of k_i' into two cases: 1) $i > i^*$, and 2) $i = i^*$. In Case 1, as all T^j for $j \in [n]$ have identical challenges in the corresponding edges in depth i, we can simply set $k_i' = k_i$. Case 2 boils down to showing that two distinct critical-round challenges s and $\hat{s}$ (used as secrets of SSS_{Γ^*}) result in a subset of decomposed trees $\{T^j\}_{j\in A}$ for some $A \in \Gamma$, where each T^j has distinct challenges $c_{i^*}^j$ and $\hat{c}_{i^*}^j$ in the corresponding edges in depth i^*. This follows from the "consistency testing" property of SSS_{Γ^*} (Appendix A.2) and the analysis is essentially identical to [CDS94].

Although the analysis of the above cases is rather simple, we will encounter some subtleties when it comes to more general cases.

Extension $k_{i^*} > 2$: First, we need to set $k'_{i^*} = (k_{i^*} - 1)^n + 1$ to retain the analysis of Case 2 when $k_{i^*} > 2$. This parameter adjustment is tight; with a set of $k_{i^*} - 1$ pairwise distinct challenges for each $j \in [n]$, there can be at most $(k_{i^*} - 1)^n$ pairwise distinct reconstructed secrets corresponding to the challenges of the composed protocol $\Pi^{\mathsf{cds}}_{\Gamma}$. Consequently, to obtain k_{i^*} pairwise distinct challenges for *some* j, we must set $k'_{i^*} > (k_{i^*} - 1)^n$. As this translates to the knowledge error including an additive term $(k_{i^*} - 1)^n/|S|$, $\Pi^{\mathsf{cds}}_{\Gamma}$ supports composition of a constant number of instances if the secret domain S coincides with the share domain $\mathcal{C}_{i^*}$. We address this limitation in Section 5.3.

Extension $i^* = i^*_{\mathsf{zk}} > 1$: In the composition illustrated in Fig. 3, the critical round i^* refers specifically to the zero-knowledge critical round i^*_{zk} relative to the CRSHVZK simulator. We show an arbitrary choice of i^*_{zk} is possible, while a careful analysis of SS parameters $\vec{k}'$ is necessary. If $i^* > 1$, the branches in round $i < i^*$ may obtain potentially different qualified sets $A_1, \ldots, A_{k'_i} \in \Gamma$ defined by k'_i sub-trees. To collect k_i pairwise distinct challenges for *all* j in a fixed qualified set A, one might naively set $k'_i = |\Gamma| \cdot k_i$ to account for every potential qualified set k_i times. However, through a more refined analysis, it can be shown that $k'_i = nk_i$ for all $i < i^*$ are sufficient.

In Theorem 3, we present the security of $\Pi^{\mathsf{cds}}_{\Gamma}$ in its most general form, encompassing all the above extensions. We then introduce a series of remarks, followed by the theorem proof.

Theorem 3 (MR-CDS Composition: $\Pi^{\mathsf{cds}}_{\Gamma}$). *For each $j \in [n]$, consider a $(2\mu+1)$-move public-coin proof protocol Π^j for relation R^j. If SSS_{Γ^*} is a semi-smooth perfect secret sharing scheme over domain S for access structure Γ^*, and every Π^j is $\vec{k}$-SS and CRSHVZK at round $i^* = i^*_{\mathsf{zk}}$, then, protocol $\Pi^{\mathsf{cds}}_{\Gamma}$ is a $(2\mu+1)$-move proof system for relation $R_\Gamma((x^1, \ldots, x^n), (w^1, \ldots, w^n))$ that outputs 1 if and only if $\exists A \in \Gamma, \forall j \in A,\ R^j(x^j, w^j) = 1$. It is complete, $\vec{k}'$-SS and CRSHVZK at round i^*, where*

1. *For $i > i^*$, $k'_i = k_i$;*
2. *For $i = i^*$, $k'_i = (k_i - 1)^n + 1$;*
3. *For $i < i^*$, $k'_i = n(k_i - 1) + 1$.*

Remark 1. This theorem gives us an interesting insight. When $k^*_i = 2$ and $i^* = 1$, we have $k'_i = k_i$ for all i, i.e., the shape of the tree preserves. Thus MR-CDS is closed for such protocols. This observation suggests that, for the purpose of composition, it is advantageous to consider the KKW protocol with $i^* = 1$ rather than $i^* = 2$ (see Theorem 2 for the critical-round flexibility of KKW).

Remark 2. Theorem 3 is also useful even if each protocol is Extended $\vec{h}$-Special Sound. The protocol Π^j for relation R^j with $\vec{h}$-ExSS on hardness assumption R^j_{hd} can be viewed as a $\vec{h}$-SS protocol for relation $\bar{R}^j$ that outputs 1 if and only if $R^j(x, w) = 1 \vee R^j_{hd}(y, u) = 1$. Therefore, we obtain a composed protocol for relation $\bar{R}_\Gamma((x^1, y^1, \ldots, x^n, y^n), (w^1, u^1, \ldots, w^n, u^n))$ that outputs 1 if and only if $\exists A \in \Gamma, \forall j \in A, \bar{R}^j((x^j, y^j), (w^j, u^j)) = 1$. This essentially means that for every instance $j \in A$, a cheating prover must know either witness w^j or a solution u^j to the hard problem. Assuming that every R^j_{hd} is computationally hard relation, one can then conclude the argument of knowledge of $\Pi^{\mathsf{cds}}_{\Gamma}$ for relation R_Γ.

Example Instantiations. We illustrate the simplest case, compressed Σ-protocols. Since $i^* = i^*_{\mathsf{zk}} = 1$ and $k_{i^*} = 2$ from Proposition 4, we can conclude $\vec{k}' = \vec{k}$ from Theorem 3 if n protocols are composed. Given Proposition 2, the composition of n $\mathsf{BP\text{-}ARP}$ protocols gives rise to CRSHVZK at round $i^*_{\mathsf{zk}} = 3$ and $\vec{k}'$-CRSS, where $\vec{k}' = (n\ell d, n(\ell + 2), 2^n + 1, 2, 7, \ldots, 7)$. Similarly, Proposition 3 reveals that the composition of n $\mathsf{BP\text{-}ACS}$ protocols gives rise to CRSHVZK at round $i^*_{\mathsf{zk}} = 3$ and $\vec{k}'$-SS, where $\vec{k}' = (n\ell, n(Q + 1), 6^n + 1, 2, 7, \ldots, 7)$.

We prove Theorem 3 below.

Proof. **Completeness.** It directly follows from the completeness and critical-round special honest verifier zero-knowledge of Π^j. We note that $\mathsf{Complete}_{\Gamma^*}$ works for the set $\bar{A}$ of shares $\{s^j\}_{j \in \bar{A}}$ since $\bar{A} \notin \Gamma^*$ when $A \in \Gamma$.

Critical-Round Special Honest Verifier Zero-Knowledge. Consider the following simulator for $\Pi^{\mathsf{cds}}_{\Gamma}$ that proceeds as follows, given as input $\{x^j\}_{j \in [n]} \in \mathcal{L}_{R_\Gamma}$, $\{c_i\}_{i \neq i^*}$ and s:

1. Compute $\{c^j_{i^*}\}_{j \in [n]} = \{s^j\}_{j \in [n]} \leftarrow \mathsf{Share}_{\Gamma^*}(s)$

2. For all $j \in [n]$: $(st_1^j, a^j) \leftarrow \mathsf{SimA}^j(x^j, c_{i^*}^j)$
3. For all $i \in [\mu - 1]$ and $j \in [n]$: $(st_{i+1}^j, z_i^j) \leftarrow \mathsf{SimZ}^j(st_i^j, c_i^j)$, where $c_i^j = c_i$ if $i \neq i^*$
4. Output $(\{a^j\}_{j\in[n]}, \{c_i\}_{i\in[\mu]\setminus\{i^*\}}, \{z_i^j\}_{i\in[\mu],j\in[n]}, s, \{s^j\}_{j\in[n]})$

The perfect hiding and share completion properties of SSS_{Γ^*} imply that the distribution of $\{s^j\}_{j\in[n]}$ generated as above is identical to that of a real transcript. Since $\{x^j\}_{j\in[n]} \in \mathcal{L}_{R_\Gamma}$, for $j \in \bar{A}$, the simulator generates a^j, z_i^j exactly as a real prover; for $j \in A$, the distribution of a^j, z_i^j output by CRSHVZK simulators for Π^j is indistinguishable with the one generated by a real prover. Grouping Step 1-2 together as SimA and Step 3 as SimZ, respectively, the above algorithm is indeed a valid CRSHVZK simulator.

As in [CDS94], if SSS_{Γ^*} additionally has smoothness, one can obtain a variant of Theorem 3 in which CRSHVZK is replaced by non-special CRHVZK. To do so, we can modify **Step** 1 of Fig. 3 so that prover instead samples independent and uniformly random shares $s^j = c_{i^*}^j$ for $j \in \bar{A}$. Clearly, this modification allows the invocation of non-special simulators for Π^j as they output uniformly random challenges in the critical round i^*. Note that the resulting simulator for $\Pi_\Gamma^{\mathsf{cds}}$ is also non-special because all challenges in round i^* must be sampled internally by the underlying non-special simulators for Π^j.

Special Soundness. The proof can be viewed as a generalization of [CDS94]. However, we must carefully choose the special soundness parameters for $\Pi_\Gamma^{\mathsf{cds}}$ in such a way that the corresponding tree of accepting transcripts contains a well-structured tree for every $j \in A \in \Gamma$ from which the extractor Ext^j for Π^j can successfully extract witness for x^j. We construct an efficient extractor $\mathsf{Ext}^{\mathsf{cds}}$ that determines some $A \in \Gamma$ and then internally runs sub-extractors $\{\mathsf{Ext}^j\}_{j\in A}$ to output a valid set of witnesses $\mathbf{w} = \{w^j\}_{j\in A}$, given as input any $\vec{k}'$-tree T^{cds} of accepting transcripts.

We first recall the structure of the tree T^{cds} for $\Pi_\Gamma^{\mathsf{cds}}$. Each path in T^{cds} defines a transcript

$$(\{a^j\}_{j\in[n]}, \{c_i\}_{i\in[\mu]\setminus\{i^*\}}, \{z_i^j\}_{i\in[\mu],j\in[n]}, s, \{s^j\}_{j\in[n]})$$

which can be decomposed into n individual transcripts:

$$\{(a^j, \{c_i\}_{i\in[\mu]\setminus\{i^*\}}, c_{i^*}^j, \{z_i^j\}_{i\in[\mu]})\}_{j\in[n]}$$

where $c_{i^*}^j = s^j$ and $c_{i^*}^j = s^j$ for $j \in [n]$. Thus, T^{cds} can also be decomposed into n individual trees, $T^1, \ldots, T^n$, where every path of T^j is derived from decomposition of the corresponding path in T^{cds}. We first observe the following properties:

1. In non-critical rounds $i \neq i^*$, the challenges in the decomposed trees T^j for $j \in [n]$ remain the same as the corresponding challenges in T^{cds}.
2. In round i^*, the shares s^j in the decomposed trees T^j for $j \in [n]$ pass the consistency check performed by $\mathsf{CheckShares}_{\Gamma^*}$ with respect to the corresponding challenge s in T^{cds}.

Case 1: $i^* < i \leq \mu$. We first prove the following statement: if $i^* < i \leq \mu$, then for every $j \in [n]$, the corresponding decomposed tree T^j has k_i pairwise distinct challenges at round i. This is straightforward: as every set of edges in depth i of T^{cds} are labeled by k_i pairwise distinct challenges, the corresponding edges in T^j have the same property.

Case 2: $i = i^*$. Next, we prove the following statement: if $k'_{i^*} = (k_{i^*} - 1)^n + 1$, then for any $(1, \ldots, 1, k'_{i^*}, k_{i^*+1}, \ldots, k_\mu)$-subtree $T_{i^*}^{\mathsf{cds}}$ for $\Pi_\Gamma^{\mathsf{cds}}$, there exists $A \in \Gamma$ such that for every $j \in A$, the corresponding decomposed subtree $T_{i^*}^j$ contains a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-subtree $\tilde{T}_{i^*}^j$ for statement x^j. We first denote by s_ℓ the ℓth challenge for round i^* in $T_{i^*}^{\mathsf{cds}}$, where s_ℓ for $\ell \in [k'_{i^*}]$ are pairwise distinct. Moreover, $\{s_\ell^j\}_{j\in[n]}$ denotes a set of shares reconstructing to s_ℓ in $T_{i^*}^{\mathsf{cds}}$. Then we have that $\mathsf{CheckShares}_{\Gamma^*}(s_\ell, \{s_\ell^j\}_{j\in[n]}) = 1$ for every $\ell \in [k'_{i^*}]$. We show that, for every $B \in \Gamma^*$, there exists $j \in B$ such that at least k_{i^*} elements of $\{s_\ell^j\}_{\ell\in[k'_{i^*}]}$ are pairwise distinct. Assume for contradiction that there exists $\tilde{B} \in \Gamma^*$ such that for every $j \in \tilde{B}$ at most $k_{i^*} - 1$ elements of $\{s_\ell^j\}_{\ell\in[k'_{i^*}]}$ are pairwise distinct. By the consistency testing property of SSS_{Γ^*} (see Appendix A.2), however, it must be that $\mathsf{Rec}_{\Gamma^*}(\{s_\ell^j\}_{j\in\tilde{B}}) = s_\ell$ for every $\ell \in [k'_{i^*}]$. As the output of Rec_{Γ^*} can take at most $(k_{i^*} - 1)^{|\tilde{B}|}$ distinct values and $|\tilde{B}| \leq n$, it contradicts the fact that $\{s_\ell\}_{\ell\in[k'_{i^*}]}$ are pairwise distinct when $k'_{i^*} = (k_{i^*} - 1)^n + 1$.

Finally, let $A = \bigcup_{B \in \Gamma^*} J_B$, where J_B denotes a set of $j \in B$ such that at least k_{i^*} elements of $\{s^j_\ell\}_{\ell \in [k'_{i^*}]}$ are pairwise distinct. As each J_B is guaranteed to be non-empty, we have that $A \cap B \neq \emptyset$ for every $B \in \Gamma^*$, which implies $A \in \Gamma$ by Proposition 5. Thus, for each $j \in A \in \Gamma$, one can derive from $T^j_{i^*}$ a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-subtree $\tilde{T}^j_{i^*}$ of accepting transcripts for statement x^j.

Case 3: $i < i^*$. We prove the following statement: if $k'_i = n(k_i - 1) + 1$ for $i < i^*$, $k'_{i^*} = (k_{i^*} - 1)^n + 1$, then for any $(1, \ldots, 1, k'_i, \ldots, k'_{i^*}, k_{i^*+1}, \ldots, k_\mu)$-subtree T^{cds}_i for $\Pi^{\mathsf{cds}}_\Gamma$, there exists $A \in \Gamma$ such that for every $j \in A$, the corresponding decomposed subtree T^j_i contains a $(1, \ldots, 1, k_i, \ldots, k_{i^*}, k_{i^*+1}, \ldots, k_\mu)$-subtree $\tilde{T}^j_i$ for statement x^j. Before delving into details, we first sketch the proof strategy for $i = i^* - 1$ (the proof for $i < i^* - 1$ can be made iteratively). By the analysis of Case 2, we have that, for each $\ell \in [k'_{i^*}]$, the ℓ-th $(1, \ldots, 1, k'_{i^*}, k_{i^*+1}, \ldots, k_\mu)$-subtree determines some access structure $A_\ell \in \Gamma$. As A_ℓ's might be potentially distinct from each other, to obtain a well-structured upper-level subtree, we need to determine some $\hat{\ell} \in [k'_{i^*-1}]$ in such a way that for every $j \in A_{\hat{\ell}}$, there are at least k_{i^*-1} subtrees that determine access structures having j as a member. We stress that the extracted subtrees $\{\tilde{T}^j_{i^*-1}\}_{j \in A_{\hat{\ell}}}$ as a result may potentially contain different sets of level $i^* - 1$ challenges for each j. For instance, if $A_{\hat{\ell}} = \{1, 2\}$ and $k_{i^*-1} = 2$, then it could be that $\tilde{T}^1_{i^*-1}$ for statement x^1 has its level $i^* - 1$ challenges labeled by c_1 and c_2, while $\tilde{T}^2_{i^*-1}$ for statement x^2 has the challenges labeled by c_3 and c_4, respectively.

More formally, we first denote by c_ℓ the ℓth challenge for round $i = i^* - 1$ in T^{cds}_i for $\ell \in [k'_i]$.[13] Then for each $\ell \in [k'_i]$, T^{cds}_i defines the corresponding $(1, \ldots, 1, k'_{i^*}, k_{i^*+1}, \ldots, k_\mu)$-subtree $T^{\mathsf{cds}}_{i^*,\ell}$ of T^{cds}_i. By the analysis of Case 2, $T^{\mathsf{cds}}_{i^*,\ell}$ defines some qualified set $A_\ell \in \Gamma$ such that for every $j \in A_\ell$, the corresponding decomposed subtree $T^j_{i^*,\ell}$ contains a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-subtree $\tilde{T}^j_{i^*,\ell}$ for statement x^j. Now we show that there exists $\hat{\ell} \in [k'_i]$ such that, for all $j \in A_{\hat{\ell}}$, the number of A_ℓ containing j is at least k_i. That is, let $L_j = \{\ell \in [k'_i] : j \in A_\ell\}$ for $j \in [n]$, we prove $\exists \hat{\ell} \in [k'_i]$ such that $\forall j \in A_{\hat{\ell}}$, $|L_j| \geq k_i$. Assume for contradiction that $\forall \hat{\ell} \in [k'_i]$, $\exists j_{\hat{\ell}} \in A_{\hat{\ell}}$ such that $|L_{j_{\hat{\ell}}}| \leq k_i - 1$. Consider the union of all such sets $L = \bigcup_{\hat{\ell} \in [k'_i]} L_{j_{\hat{\ell}}}$. Observe that for each $\hat{\ell} \in [k'_i]$ and $j_{\hat{\ell}} \in A_{\hat{\ell}}$, we have $\hat{\ell} \in L_{j_{\hat{\ell}}}$. Thus, it must be that $|L| \geq k'_i = n(k_i - 1) + 1$. However, since each $L_{j_{\hat{\ell}}}$ has cardinality at most $k_i - 1$, we have $|L| = |\bigcup_{\hat{\ell} \in [k'_i]} L_{j_{\hat{\ell}}}| \leq |\bigcup_{j_{\hat{\ell}} \in [n]} L_{j_{\hat{\ell}}}| \leq n(k_i - 1)$, which leads to contradiction.

Since each A_ℓ is associated with pairwise distinct c_ℓ's, we obtain a $(1, \ldots, k_{i^*-1},\ k_{i^*}, \ldots, k_\mu)$-subtree $\tilde{T}^j_i$ for every $j \in A_{\hat{\ell}}$. Applying the above argument iteratively for $i = i^* - 1, \ldots, 1$, we prove the statement for Case 3.

Extractor. Combining Case 1-3 above, there exists $A \in \Gamma$ such that for every $j \in A$, the corresponding decomposed tree T^j contains a $\vec{k}$-tree $\tilde{T}^j$ for Π^j. The extractor $\mathsf{Ext}^{\mathsf{cds}}$ then simply runs sub-extractors Ext^j on $\tilde{T}^j$ to extract witnesses for all $j \in A$. As Π^j is $\vec{k}$-SS, every Ext^j always succeeds in extracting witness. □

5.3 Suppressing Exponential Blow-up via Multi-Round Share-then-Hash

To support polynomial-size compositions, we generalize the Share-then-Hash method with the help of a collision resistant hash function (CRH) $H : \{0,1\}^* \mapsto \mathcal{C}_{i^*}$. We present MR-StH composition, $\Pi^{\mathsf{sth}}_\Gamma$, formally in Fig. 3 alongside $\Pi^{\mathsf{cds}}_\Gamma$. The only modification is that the prover is now required to obtain challenges $c^j_{i^*} \leftarrow H(\ldots, s^j)$, instead of setting $c^j_{i^*} = s^j$. Unlike the CDS composition, MR-StH allows for a secret sharing scheme with larger domains than the challenge space $\mathcal{C}_{i^*}$. Thus, one can compensate for the exponential blowup in the numerator by setting $|S| \gg (k_{i^*} - 1)^n$. As sketched in Section 2.2, the knowledge soundness of $\Pi^{\mathsf{sth}}_\Gamma$ is analyzed within the predicate special soundness framework of [AAB+24] (recalled in Appendix B.1), instead of the standard special soundness notion. We state the following theorem whose formal analysis is deferred to Appendix B.

Theorem 4 (MR-StH Composition: $\Pi^{\mathsf{sth}}_\Gamma$). *For each $j \in [n]$, consider a $(2\mu + 1)$-move public-coin proof protocol Π^j for relation R^j. If SSS_{Γ^*} is a semi-smooth perfect secret sharing scheme over domain S for access structure Γ^*, $H : \{0,1\}^* \mapsto \mathcal{C}_{i^*}$ is a collision-resistant hash function, and every*

[13] Note that we are abusing the notation here, because the ℓth challenge for round i should be technically denoted by $c_{i,\ell}$ following the protocol description. We drop the subscript i for readability since we focus on the ith round when proving Case 3.

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 $$c_i := \begin{cases} H_i(x, a, z_1, \ldots, z_{i-1}) & (i < i^*_{\mathsf{zk}}) \\ m & (i = i^*_{\mathsf{zk}}) \\ H_i(x, a, m, z_1, \ldots, z_{i-1}) & (i > i^*_{\mathsf{zk}}). \end{cases} \quad (1)$$ 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 (1). Output $com := (a, c_1, z_1, \ldots, c_{i^*_{\mathsf{zk}}-1}, z_{i^*_{\mathsf{zk}}-1})$ 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 (1). 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 (1) and output $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.

Fig. 4. Trapdoor commitment scheme $\mathsf{K}_{\Pi_{i^*_{\mathsf{zk}}}}$.

Π^j is $\vec{k}$-SS and CRSHVZK at round $i^* = i^*_{\mathsf{zk}}$, then, protocol $\Pi^{\mathsf{sth}}_\Gamma$ is a $(2\mu+1)$-move proof system for relation $R_\Gamma((x^1, \ldots, x^n), (w^1, \ldots, w^n))$ that outputs 1 if and only if $\exists A \in \Gamma, \forall j \in A,\ R^j(x^j, w^j) = 1$. It is complete, $(\vec{k}', \boldsymbol{\Phi})$-predicate-special-sound for $\vec{k}' = (k'_1, \ldots, k'_\mu)$ and a predicate system $\boldsymbol{\Phi}$, and CRSHVZK at round i^*, where

1. *For $i > i^*$, $k'_i = k_i$ and $\Phi^{\mathsf{chal}}_{i,\ell}$ for each $\ell \in [k'_i]$ always outputs 1;*
2. *For $i = i^*$, $k'_i = \tau \geq n(k_i - 2) + 2$ and $\Phi^{\mathsf{chal}}_{i,\ell}(t_1, \ldots, t_{\ell-1}, s_\ell)$ for each $\ell \in [k'_i]$ return 1 if (2) is satisfied, and 0 otherwise;*
3. *For $i < i^*$, $k'_i = n(k_i - 1) + 1$ and $\Phi^{\mathsf{chal}}_{i,\ell}$ for $\ell \in [k'_i]$ always outputs 1.*

Plugging Theorem 4 and the analysis of *failure density* (Lemma 2) into the general theorem on Fiat-Shamir knowledge soundness established by [AAB+24] (recalled in Theorem 6) results in the following corollary. The last term in the knowledge error essentially quantifies the probability that a uniformly sampled $s \in S$ (i.e., the i^*_{zk}-th round challenge) falls into a "bad" set of secrets that could cause extraction to fail.

Corollary 1. *The Fiat-Shamir-transformed version of $\Pi^{\mathsf{sth}}_\Gamma$ is adaptively knowledge sound in the random oracle model with knowledge error*

$$2(Q+1)\left(\sum_{i \in [\mu] \setminus \{i^*_{\mathsf{zk}}\}} \frac{k'_i - 1}{|\mathcal{C}_i|} + \frac{(k'_{i^*_{\mathsf{zk}}} - 1) \cdot 2^n \cdot (k_{i^*_{\mathsf{zk}}} - 1)^n}{|S|}\right)$$

where Q is the number of random oracle queries made by a cheating prover.

6 Trapdoor Commitment from Multi-Round Protocol

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 $H_i : \{0,1\}^* \to \mathcal{C}_i$, we construct a trapdoor commitment scheme $\mathsf{K} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{Equiv}, \mathsf{Ver})$ as illustrated in Figure 4. For simplicity, we assume that the message space $\mathcal{M} = \mathcal{C}_{i^*_{\mathsf{zk}}}$.

Theorem 5. *$\mathsf{K}_{\Pi_{i^*_{\mathsf{zk}}}}$ in Figure 4 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 and trapdoor extractable if*
  - *$\Pi_{i^*_{\mathsf{zk}}}$ is perfectly complete,*
  - *$\Pi_{i^*_{\mathsf{zk}}}$ is (extended) $\vec{k}$-SS with $k_{i^*_{\mathsf{zk}}} = 2$,*

- *Relation R is one-way,*
- *H_i is a programmable random oracle.*

– *It is offline trapdoor extractable if, additionally, $\Pi_{i^*_{\mathsf{zk}}}$ is $\vec{k}'$-CRSS with $i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}}$, $k_{i^*_{\mathsf{zk}}} = k'_{i^*_{\mathsf{zk}}}$, and $k'_i = 1$ for all $i \in [\mu] \setminus i^*_{\mathsf{zk}}$.*

Proof. (*Correctness*) It holds directly from the perfect completeness of $\Pi_{i^*_{\mathsf{zk}}}$ and the critical-round zero-knowledge at round i^*_{zk}.

(*Hiding Property*) It is proved by a game transition argument. We begin with the left term of the hiding definition in Definition 6;

$$\begin{aligned} P_0 &:= \Pr\left[1 \leftarrow \mathcal{A}(ck, com) \,|\, (ck, td) \leftarrow \mathsf{Gen}(1^\lambda), (com, open) \leftarrow \mathsf{Com}(ck, m)\right] \\ &= \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})) & \begin{array}{l}(x, w) \leftarrow L_{RW}(1^\lambda), \\ (st_1, a) \leftarrow \mathsf{SimA}(x, m), \\ \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)\end{array} \end{array}\right] \end{aligned}$$

where c_i is computed following (1). We then modify the game by replacing the zero-knowledge simulator with a real prover algorithm.

$$P_1 := \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})) & \begin{array}{l}(x, w) \leftarrow L_{RW}(1^\lambda), \\ (st_1, a) \leftarrow \mathsf{A}(x, w), \\ \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i)\end{array} \end{array}\right]$$

Then $|P_0 - P_1|$ is bound by the zero-knowledge error, say ϵ_{crzk}, as in Definition 8. Finally, we make it back to the simulation as follows.

$$P_2 = \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})) & \begin{array}{l}(x, w) \leftarrow L_{RW}(1^\lambda), \\ (st_1, a) \leftarrow \mathsf{SimA}(x, m'), \\ \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)\end{array} \end{array}\right]$$

Again, $|P_1 - P_2|$ is upper bound by ϵ_{crzk}. Since P_2 is the same as the right term of the hiding definition, we obtain $P_0 - P_2 \leq 2\epsilon_{\mathsf{crzk}}$ as a bound for the hiding property that is negligible as ϵ_{crzk} is negligible by assumption.

(*Equivocability*) It holds due to CRSHVZK of $\Pi_{i^*_{\mathsf{zk}}}$ since the equivocability game in Definition 6 is exactly the same as the CRSHVZK game in Definition 8.

(*Binding and Trapdoor Extractability*) Let $\mathcal{A}$ be a successful adversary that opens a commitment in two ways with probability ϵ_{bin}. Precisely, given the commitment key x, it outputs $(com, m_0, open_0, m_1, open_1)$ such that $m_0 \neq m_1$, $m_0, m_1 \in \mathcal{C}_{i^*_{\mathsf{zk}}}$, and $\mathsf{Ver}(com, m_j, open_j) = 1$ for $j \in \{0, 1\}$. For simplicity, we assume that all the challenge spaces $\mathcal{C}_i$ are identical and represent each H_i using a single random oracle H. Overall structure of the proof follows that of the Fiat-Shamir security originally from [AFK23].

Given $\mathcal{A}$ as a black-box, we construct $\mathcal{B}$ that breaks the one-wayness of R. It constructs a $\vec{k}$-tree of transcripts by running the tree builder $\mathcal{T}_0$, as illustrated in Figure 5. $\mathcal{B}$ simulates H by lazy sampling. Whenever H receives a fresh query, $\mathcal{B}$ returns a random value. By $(in, out) \to H$, we denote programming H as $out = H(in)$. If $\mathcal{T}_0$ outputs $v = 1$ with a tree of transcripts tr_0, $\mathcal{B}$ gives tr_0 to the extractor. Although tr_0 will contain challenges at round i^*_{zk}, i.e., messages chosen by $\mathcal{A}$, the extractor Ext outputs a witness (or ExExt may output a solution to the hard problem instead) by definition of (extended) $\vec{k}$-SS regardless of what the messages are as long as they are pairwise distinct to form a valid tree for $k_{i^*_{\mathsf{zk}}} = 2$.

The recursive tree builder $\mathcal{T}_i$ is essentially the same as the one in [AFK23] for the multi-round Fiat-Shamir security. However, there are notable differences due to the adversary $\mathcal{A}$ playing the binding game rather than the standard Fiat-Shamir security game as in [AFK23].

– Parallel Tree Building: $\mathcal{A}$ outputs two transcripts, $(com, m_0, open_0)$ and $(com, m_1, open_1)$, branching at round i^*_{zk}. Both of them must be included in the tree since the self-chosen values m_0 and m_1 could represent the only challenges at round i^*_{zk} for which $\mathcal{A}$ is successful. Therefore, $\mathcal{T}_{i>i^*_{\mathsf{zk}}}$ must construct sub-trees for both branches (com, m_0) and (com, m_1) in parallel, as described in Step 2 of the case $i > i^*_{\mathsf{zk}}$ in Figure 5. Within each sub-tree construction, at most one of the sub-trees returned from the descending builder $\mathcal{T}_{i+1}$ will match the intended branch.
$\mathcal{T}_{i^*_{\mathsf{zk}}}$ only needs to combine the two sub-trees into a single structure. The ancestor builders $\mathcal{T}_{i<i^*_{\mathsf{zk}}-1}$ remain consistent with the original builders outlined in [AFK23].

(In case of abort, return with $v = 0$.)

Case $0 \leq i < i^*_{\mathsf{zk}}$:

1. Run $\mathcal{T}_{i+1} \to (I, tr_{i+1}, v)$. Abort if $v = 0$.
2. Initialize $used := \{H(I_i)\}$ and $tr_i := \{tr_{i+1}\}$.
3. Repeat the following until $|tr_i| = k_i$ happens:
   - If $|used| = |\mathcal{C}_i|$, abort.
   - Sample $c \leftarrow \mathcal{C}_i \setminus used$, and program as $(I_i, c) \to H$. Update $used \leftarrow used \cup \{c\}$. Run $\mathcal{T}_{i+1} \to (I', tr'_{i+1}, v')$.
   - If $I'_i = I_i$ and $v' = 1$, update $tr_i \leftarrow tr_i \cup tr'_{i+1}$.
4. Return $(I, tr_i, 1)$.

Case $i = i^*_{\mathsf{zk}}$:

1. Run $\mathcal{T}_{i+1} \to (I^{(0)}, tr^{(0)}_{i+1}, I^{(1)}, tr^{(1)}_{i+1}, v)$.
2. Set $I := I^{(0)}$ and $tr_i := (tr^{(0)}_{i+1}, {tr_{i+1}}^{(1)})$.
3. Return (I, tr_i, v).

Case $i^*_{\mathsf{zk}} < i < \mu$:

1. Run $\mathcal{T}_{i+1} \to (I^{(0)}, tr^{(0)}_{i+1}, I^{(1)}, tr^{(1)}_{i+1}, v)$. Abort if $v = 0$.
2. Do this step for $b = \{0, 1\}$:
   - Initialize $used := \{H(I^{(b)}_i)\}$ and $tr^{(b)}_i := \{tr^{(b)}_{i+1}\}$.
   - Repeat the following until $|tr^{(b)}_i| = k_i$ happens:
     * If $|used| = |\mathcal{C}_i|$, abort.
     * Sample $c \leftarrow \mathcal{C}_i \setminus used$, and program as $(I_i, c) \to O$. Update $used \leftarrow used \cup \{c\}$. Run $\mathcal{T}_{i+1} \to ({I'}^{(0)}, {tr'}^{(0)}_{i+1}, {I'}^{(1)}, {tr'}^{(1)}_{i+1}, v')$.
     * If $v' = 1$ and $I^{(b)}_i = {I'}^{(b')}_i$ for either $b' = 0$ or $b' = 1$, update $tr^{(b)}_i \leftarrow tr^{(b)}_i \cup {tr'}^{(b')}_{i+1}$.
3. Return $(I^{(0)}, tr^{(0)}_i, I^{(1)}, tr^{(1)}_i, 1)$.

Case $i = \mu$:

1. Run $\mathcal{A} \to (com, m_0, open_0, m_1, open_1)$. If $\mathsf{Ver}(x, com, m_0, open_0) = 1$, $\mathsf{Ver}(x, com, m_1, open_1) = 1$, and $m_0 \neq m_1$, set $v = 1$. Set $v = 0$, otherwise.
2. For $com = (a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$ and $open_0 = (z_{i^*_{\mathsf{zk}}}, \ldots, z_\mu)$, let $I^{(0)}_0 = (a)$ and $I^{(0)}_j = (a, z_1, \ldots, z_j)$ for $j = 1, \ldots, \mu$. Define $I^{(0)} := (I^{(0)}_0, \ldots, I^{(0)}_\mu)$ and $tr^{(0)}_\mu := (com, m_0, open_0)$. Define $I^{(1)}$ and $tr^{(1)}_\mu$ in the same manner regarding com and $open_1$.
3. Return $(I^{(0)}, tr^{(0)}_\mu, I^{(1)}, tr^{(1)}_\mu, v)$.

Fig. 5. Tree builder $\mathcal{T}_i$ with binding adversary $\mathcal{A}$.

– Self-Chosen Challenges: In round i^*_{zk}, the challenges m_0 and m_1 are selected by $\mathcal{A}$. But this is not essentially different from chosing them uniformly and consider the case that $\mathcal{A}$ succeeds. The process of constructing the sub-tree for $i > i^*_{\mathsf{zk}}$ is analogous to allowing the adversary to choose the initial message a in the top tree. Consequently, the challenge m is included as a part of input to $H_{i>i^*_{\mathsf{zk}}}$. If $\mathcal{A}$ selects m_0 and m_1 such that tree building fails, it subsequently reduces the success probability of $\mathcal{A}$, as demonstrated in Proposition 2 of [AFK23]. Furthermore, since the sub-extractor $\mathcal{T}_{i^*_{\mathsf{zk}}}$ succeeds with the same probability as $\mathcal{T}_{i^*_{\mathsf{zk}}-1}$, it can be considered merged into $\mathcal{T}_{i^*_{\mathsf{zk}}-1}$ when analyzing the bounds.

From Proposition 2 in [AFK23], the probability that $\mathcal{T}_0$ returns $v = 1$ is at least

$$\frac{\epsilon_{\mathsf{bin}} - (Q+1)\theta}{1-\theta}$$

where $\theta = 1 - \prod_{i \in [\mu] \setminus i^*_{\mathsf{zk}}} (1 - \frac{k_i - 1}{|\mathcal{C}_i|})$. The expected number of invocations to $\mathcal{A}$ is at most $\xi + Q \cdot (\xi - 1)$ where $\xi = \prod_{i \in [\mu] \setminus i^*_{\mathsf{zk}}} k_i$.

Since the above actually extracts a witness (in the case of extended SS, under the relevant hardness assumption), the trapdoor extractability is proved as well.

(*Offline Trapdoor Extractability*) First observe that the adversary here is the same as the one in the binding game. Then, recall that we assumed that $\Pi_{i^*_{\mathsf{zk}}}$ is CRSS whose tree of transcripts where i^*_{zk} is the

first and the only branching round. Therefore, a valid collision $(com, m_0, open_0, m_1, open_1)$ constitutes a CRSS tree of transcript. As it is obtained by running the adversary with uniformly chosen challenges, the resulting CRSS tree is a uniformly sampled one from the distribution of subtrees of the extended SS trees built from the binding-game adversary where the challenges in the i^*_{zk} are chosen by the adversary. Then, from Theorem 1, we can conclude that the given extractor CrExt outputs a witness on the tree except for a negligible probability.

□

In summary, TDC is realizable if $k_{i^*_{\mathsf{zk}}} \leq 2$ no matter where i^*_{ss} is, and it is additionally offline extractable if the CRSS tree of transcripts satisfies $i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}}$ and i^*_{zk} is the only branching round. KKW fits to these conditions and we can instantiate our TDC from it. We expect that natural MPCitH variations can also be seen as $i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}} = \mu$.

7 Applications to Advanced Signatures

This section presents additional applications to advanced signatures, other applications including accumulators and trapdoor commitments with flexible trapdoor allocation are found in Section E. While they can almost be seamlessly derived from the previous sections and existing generic constructions, it is worthwhile to mention them in order to illustrate the significant influence that critical-round protocols exert on cryptographic protocol design.

Previous works [HS20,LTZ24] pose the open question of constructing post-quantum trapdoor commitments for adaptor signatures (AS) and threshold ring signatures (TRS) without the overhead of generic NP reductions, specifically asking about instantiations from MPCitH. We resolve this by instantiating their generic compilers with our (standard) trapdoor commitment from Theorem 5, which meets all required properties and can be built from MPCitH via Theorem 2.

7.1 Adaptor signatures

Adaptor signatures were introduced by Poelstra [Poe17] and formalized in [AEE+21] [DOY22,GSST24], as a useful trick available to Schnorr signatures. An adaptor signature scheme is defined with respect to a one-way NP relation $\mathcal{L}_{RW}$ with instance-witness pairs $(x, w) \in \mathcal{L}_{RW} \iff R(x, w) = 1$, and a digital signature scheme. A signer who holds a secret key sk can pre-sign a message m with respect to instance x, obtaining a so-called pre-signature $\bar{\sigma}$. Later, $\bar{\sigma}$ can be adapted to a standard signature σ by some party that knows the witness w. Finally, from a pre-signature $\bar{\sigma}$ and the signature σ, it is possible to extract a valid witness w such that $R(x, w) = 1$.

A recent work by Liu, Tzannetos, and Zikas [LTZ24] introduces a generic construction of witness-hiding AS from any digital signature scheme and any hiding and offline trapdoor extractable trapdoor commitment scheme. Their AS achieves post-quantum security if both the signature scheme and the trapdoor commitment are also post-quantum secure. They instantiate the latter[14] for any NP relation via the three-move interactive protocol for graph hamiltonicity, at the cost of a large practical overhead.

Applying Theorem 2 to the compiler in [LTZ24] actually yields adaptor signatures from MPC in the head with reasonable practical parameters. Signature size is dominated by the MPCitH protocol for the one-way function that parametrizes the trapdoor commitment, and by the size of a standard signature scheme. For example, if the one-way relation is instantiated by a block cipher such as LowMC [ARS+15], an instantiation building upon the Picnic signature scheme [CDG+20,KZ20] yields adaptor signature sizes on the order of 30 KB (for the lowest security level).

7.2 Threshold ring signatures

(t, N)-threshold ring signatures (TRS) [BSS02] allow a set of t signers among a "ring" of N participants to jointly produce a signature σ on a message m. Given a ring of users $R = \{R_1, \ldots, R_N\}$ where each R_i owns a pair of keys (sk_i, pk_i), TRS guarantee that 1) the t signers remain *anonymous* within the

[14] More precisely, they instantiate a weaker variant of trapdoor commitments that they call *trapdoor commitments with specific adaptable message* (TC-am), and which they show to be sufficient for their application.

set of public keys $(pk_1, \ldots, pk_N)$, and 2) if an adversary corrupts less than t parties from the ring, then it is hard for the adversary to forge a valid signature σ.

The work of Haque and Scafuro [HS20] builds TRS from any trapdoor commitment and Shamir's secret sharing. The trapdoor commitment needs to satisfy binding, hiding, and trapdoor indistinguishability (which is in turn implied by our notion of equivocability). As in the previous case, they achieve a post-quantum secure TRS by instantiating their trapdoor commitment from the interactive protocol for graph hamiltonicity.

8 Conclusion

Based on the observation that the roles of each round differ in multi-round protocols, we introduced the novel notions of CRZK and CRSS and demonstrated their usefulness. Several open questions remain:

- We observed that some multi-round proof protocols that utilize the witness in only one round can admit a CRZK simulator. Could this observation be extended into a more general theorem?
- Are the blow-ups of k'_i for rounds $i \leq i^*$ inherent in MR-CDS and MR-StH compositions in Section 5?

Acknowledgements

This work was partially done while Dung Bui and David Balbás were doing an internship at NTT Social Informatics Laboratories.
David Balbás is supported by the PICOCRYPT project that has received funding from the European Research Council (ERC) under the European Unions 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, and partially funded by Ministerio de Universidades (FPU21/00600).
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+20. M. Abe, M. Ambrona, A. Bogdanov, M. Ohkubo, and A. Rosen. Non-interactive composition of sigma-protocols via share-then-hash. In *Advances in Cryptology – ASIACRYPT 2020, Part III*, *Lecture Notes in Computer Science* 12493, pages 749–773, Daejeon, South Korea, December 7–11, 2020. Springer, Cham, Switzerland.

AAB+21. M. Abe, M. Ambrona, A. Bogdanov, M. Ohkubo, and A. Rosen. Acyclicity programming for sigma-protocols. In *TCC 2021: 19th Theory of Cryptography Conference, Part I*, *Lecture Notes in Computer Science* 13042, pages 435–465, Raleigh, NC, USA, November 8–11, 2021. Springer, Cham, Switzerland.

AAB+24. M. A. Aardal, D. F. Aranha, K. Boudgoust, S. Kolby, and A. Takahashi. Aggregating falcon signatures with LaBRADOR. In *Advances in Cryptology – CRYPTO 2024, Part I*, *Lecture Notes in Computer Science* 14920, pages 71–106, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

AABN02. M. Abdalla, J. H. An, M. Bellare, and C. Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In *Advances in Cryptology – EUROCRYPT 2002*, *Lecture Notes in Computer Science* 2332, pages 418–433, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Heidelberg, Germany.

ABF+24. G. Avitabile, V. Botta, D. Friolo, D. Venturi, and I. Visconti. Compact proofs of partial knowledge for overlapping CNF formulae. To appear in Journal of Cryptology, 2024.

ABFV22. G. Avitabile, V. Botta, D. Friolo, and I. Visconti. Efficient proofs of knowledge for threshold relations. In *ESORICS 2022: 27th European Symposium on Research in Computer Security, Part III*, *Lecture Notes in Computer Science* 13556, pages 42–62, Copenhagen, Denmark, September 26–30, 2022. Springer, Cham, Switzerland.

ABO+24. M. Abe, A. Bogdanov, M. Ohkubo, A. Rosen, Z. Shang, and M. Tibouchi. CDS composition of multi-round protocols. In *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IX*, *Lecture Notes in Computer Science* 14928, pages 391–423. Springer, 2024.

ACF21. T. Attema, R. Cramer, and S. Fehr. Compressing proofs of k-out-of-n partial knowledge. In *Advances in Cryptology – CRYPTO 2021, Part IV*, *Lecture Notes in Computer Science* 12828, pages 65–91, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland.

ACK21. T. Attema, R. Cramer, and L. Kohl. A compressed Σ-protocol theory for lattices. In *Advances in Cryptology – CRYPTO 2021, Part II*, *Lecture Notes in Computer Science* 12826, pages 549–579, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland.

AEE+21. L. Aumayr, O. Ersoy, A. Erwig, S. Faust, K. Hostáková, M. Maffei, P. Moreno-Sanchez, and S. Riahi. Generalized channels from limited blockchain scripts and adaptor signatures. In *Advances in Cryptology – ASIACRYPT 2021, Part II*, *Lecture Notes in Computer Science* 13091, pages 635–664, Singapore, December 6–10, 2021. Springer, Cham, Switzerland.

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

AFK22. T. Attema, S. Fehr, and M. Klooß. Fiat-shamir transformation of multi-round interactive proofs. In *TCC 2022: 20th Theory of Cryptography Conference, Part I*, *Lecture Notes in Computer Science* 13747, pages 113–142, Chicago, IL, USA, November 7–10, 2022. Springer, Cham, Switzerland.

AFK23. T. Attema, S. Fehr, and M. Klooß. Fiat-shamir transformation of multi-round interactive proofs (extended version). *Journal of Cryptology*, 36(4):36, October 2023.

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

AKLY24. T. Attema, M. KlooSS, R. W. F. Lai, and P. Yatsyna. Adaptive special soundness: Improved knowledge extraction by adaptive useful challenge sampling. Cryptology ePrint Archive, Paper 2024/2038, 2024.

ARS+15. M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. In *Advances in Cryptology – EUROCRYPT 2015, Part I*, *Lecture Notes in Computer Science* 9056, pages 430–454, Sofia, Bulgaria, April 26–30, 2015. Springer, Berlin, Heidelberg, Germany.

BBB+18. B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. 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.

BCC+16. J. Bootle, A. Cerulli, P. Chaidos, J. Groth, and C. Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *Advances in Cryptology – EUROCRYPT 2016, Part II*, *Lecture Notes in Computer Science* 9666, pages 327–357, Vienna, Austria, May 8–12, 2016. Springer, Berlin, Heidelberg, Germany.

BCS16. E. Ben-Sasson, A. Chiesa, and N. Spooner. Interactive oracle proofs. In *TCC 2016-B: 14th Theory of Cryptography Conference, Part II*, *Lecture Notes in Computer Science* 9986, pages 31–60, Beijing, China, October 31 – November 3, 2016. Springer, Berlin, Heidelberg, Germany.

Bd94. J. C. Benaloh and M. de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In *Advances in Cryptology – EUROCRYPT'93*, *Lecture Notes in Computer Science* 765, pages 274–285, Lofthus, Norway, May 23–27, 1994. Springer, Berlin, Heidelberg, Germany.

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

BFS20. B. Bünz, B. Fisch, and A. Szepieniec. Transparent SNARKs from DARK compilers. In *Advances in Cryptology – EUROCRYPT 2020, Part I*, *Lecture Notes in Computer Science* 12105, pages 677–706, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland.

BGTZ23. A. R. Block, A. Garreta, P. R. Tiwari, and M. Zając. On soundness notions for interactive oracle proofs. Cryptology ePrint Archive, Report 2023/1256, 2023.

BL90. J. C. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptology – CRYPTO'88*, *Lecture Notes in Computer Science* 403, pages 27–35, Santa Barbara, CA, USA, August 21–25, 1990. Springer, New York, USA.

Blu86. M. Blum. How to prove a theorem so no one else can claim it. In *The International Congress of Mathematicians (ICM), 1986*, 1986.

BN20. C. Baum and A. Nof. Concretely-efficient zero-knowledge arguments for arithmetic circuits and their application to lattice-based cryptography. In *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I*, *Lecture Notes in Computer Science* 12110, pages 495–526, Edinburgh, UK, May 4–7, 2020. Springer, Cham, Switzerland.

BP97. N. Bari and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology – EUROCRYPT'97*, *Lecture Notes in Computer Science* 1233, pages 480–494, Konstanz, Germany, May 11–15, 1997. Springer, Berlin, Heidelberg, Germany.

BSS02. E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *Advances in Cryptology – CRYPTO 2002*, *Lecture Notes in Computer Science* 2442, pages 465–480, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Berlin, Heidelberg, Germany.

CCH+19. R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. N. Rothblum, R. D. Rothblum, and D. Wichs. Fiat-Shamir: from practice to theory. In *51st Annual ACM Symposium on Theory of Computing*, pages 1082–1090, Phoenix, AZ, USA, June 23–26, 2019. ACM Press.

CDG+20. M. Chase, D. Derler, S. Goldfeder, J. Katz, V. Kolesnikov, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, X. Wang, et al. The picnic signature scheme. *Submission to NIST Post-Quantum Cryptography project*, 2020.

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

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

CHM+20. A. Chiesa, Y. Hu, M. Maller, P. Mishra, P. Vesely, and N. P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *Advances in Cryptology – EUROCRYPT 2020, Part I*, *Lecture Notes in Computer Science* 12105, pages 738–768, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland.

CLTZ24. M. Ciampi1, X. Liu, I. Tzannetos, and V. Zikas. Universal adaptor signatures from blackbox multi-party computation. Cryptology ePrint Archive, Paper 2024/1773, 2024.

CPS+16a. M. Ciampi, G. Persiano, A. Scafuro, L. Siniscalchi, and I. Visconti. Improved OR-composition of sigma-protocols. In *TCC 2016-A: 13th Theory of Cryptography Conference, Part II*, *Lecture Notes in Computer Science* 9563, pages 112–141, Tel Aviv, Israel, January 10–13, 2016. Springer, Berlin, Heidelberg, Germany.

CPS+16b. M. Ciampi, G. Persiano, A. Scafuro, L. Siniscalchi, and I. Visconti. Online/offline OR composition of sigma protocols. In *Advances in Cryptology – EUROCRYPT 2016, Part II*, *Lecture Notes in Computer Science* 9666, pages 63–92, Vienna, Austria, May 8–12, 2016. Springer, Berlin, Heidelberg, Germany.

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

CV05. D. Catalano and I. Visconti. Hybrid trapdoor commitments and their applications. In *ICALP 2005: 32nd International Colloquium on Automata, Languages and Programming*, *Lecture Notes in Computer Science* 3580, pages 298–310, Lisbon, Portugal, July 11–15, 2005. Springer, Berlin, Heidelberg, Germany.

Dam90. I. Damgård. On the existence of bit commitment schemes and zero-knowledge proofs. In *Advances in Cryptology – CRYPTO'89*, *Lecture Notes in Computer Science* 435, pages 17–27, Santa Barbara, CA, USA, August 20–24, 1990. Springer, New York, USA.

DFMS22. J. Don, S. Fehr, C. Majenz, and C. Schaffner. Online-extractability in the quantum random-oracle model. In *Advances in Cryptology – EUROCRYPT 2022, Part III*, *Lecture Notes in Computer Science* 13277, pages 677–706, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland.

DG23. Q. Dao and P. Grubbs. Spartan and bulletproofs are simulation-extractable (for free!). In *Advances in Cryptology – EUROCRYPT 2023, Part II*, *Lecture Notes in Computer Science* 14005, pages 531–562, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

DOY22. W. Dai, T. Okamoto, and G. Yamamoto. Stronger security and generic constructions for adaptor signatures. In *Progress in Cryptology - INDOCRYPT 2022: 23rd International Conference in Cryptology in India*, *Lecture Notes in Computer Science* 13774, pages 52–77, Kolkata, India, December 11–14, 2022. Springer, Cham, Switzerland.

FGQ+23. P.-A. Fouque, A. Georgescu, C. Qian, A. Roux-Langlois, and W. Wen. A generic transform from multi-round interactive proof to NIZK. In *PKC 2023: 26th International Conference on Theory and Practice of Public Key Cryptography, Part II*, *Lecture Notes in Computer Science* 13941, pages 461–481, Atlanta, GA, USA, May 7–10, 2023. Springer, Cham, Switzerland.

FHJ20. M. Fischlin, P. Harasser, and C. Janson. Signatures from sequential-OR proofs. In *Advances in Cryptology – EUROCRYPT 2020, Part III*, *Lecture Notes in Computer Science* 12107, pages 212–244, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland.

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

FJR22. T. Feneuil, A. Joux, and M. Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. In *Advances in Cryptology – CRYPTO 2022, Part II*, *Lecture Notes in Computer Science* 13508, pages 541–572, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

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

GGHAK22. A. Goel, M. Green, M. Hall-Andersen, and G. Kaptchuk. Stacking sigmas: A framework to compose Σ-protocols for disjunctions. In *Advances in Cryptology – EUROCRYPT 2022, Part II*, *Lecture Notes in Computer Science* 13276, pages 458–487, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland.

GHAKS23. A. Goel, M. Hall-Andersen, G. Kaptchuk, and N. Spooner. Speed-stacking: Fast sublinear zero-knowledge proofs for disjunctions. In *Advances in Cryptology – EUROCRYPT 2023, Part II*, *Lecture Notes in Computer Science* 14005, pages 347–378, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

GKK+22. C. Ganesh, H. Khoshakhlagh, M. Kohlweiss, A. Nitulescu, and M. Zajac. What makes fiat-shamir zkSNARKs (updatable SRS) simulation extractable? In *SCN 22: 13th International Conference on Security in Communication Networks*, *Lecture Notes in Computer Science* 13409, pages 735–760, Amalfi, Italy, September 12–14, 2022. Springer, Cham, Switzerland.

GMW91. O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

GOP+22. C. Ganesh, C. Orlandi, M. Pancholi, A. Takahashi, and D. Tschudi. Fiat-shamir bulletproofs are non-malleable (in the algebraic group model). In *Advances in Cryptology – EUROCRYPT 2022, Part II*, *Lecture Notes in Computer Science* 13276, pages 397–426, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland.

GOP+25. C. Ganesh, C. Orlandi, M. Pancholi, A. Takahashi, and D. Tschudi. Fiat-shamir bulletproofs are non-malleable (in the random oracle model). *J. Cryptol.*, 38(1):11, 2025.

GSST24. P. Gerhart, D. Schröder, P. Soni, and S. A. K. Thyagarajan. Foundations of adaptor signatures. In *Advances in Cryptology – EUROCRYPT 2024, Part II*, *Lecture Notes in Computer Science* 14652, pages 161–189, Zurich, Switzerland, May 26–30, 2024. Springer, Cham, Switzerland.

GW11. C. Gentry and D. Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *43rd Annual ACM Symposium on Theory of Computing*, pages 99–108, San Jose, CA, USA, June 6–8, 2011. ACM Press.

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

HS20. A. Haque and A. Scafuro. Threshold ring signatures: New definitions and post-quantum security. In *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part II*, *Lecture Notes in Computer Science* 12111, pages 423–452, Edinburgh, UK, May 4–7, 2020. Springer, Cham, Switzerland.

HV20. C. Hazay and M. Venkitasubramaniam. On the power of secure two-party computation. *Journal of Cryptology*, 33(1):271–318, January 2020.

IKOS07. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge from secure multiparty computation. In *39th Annual ACM Symposium on Theory of Computing*, pages 21–30, San Diego, CA, USA, June 11–13, 2007. ACM Press.

KKW18. J. Katz, V. Kolesnikov, and X. Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 525–537, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

KLP22. A. Kim, X. Liang, and O. Pandey. A new approach to efficient non-malleable zero-knowledge. In *Advances in Cryptology – CRYPTO 2022, Part IV*, *Lecture Notes in Computer Science* 13510, pages 389–418, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

Kol24. D. Kolonelos. *Succinct Cryptographic Commitments with Fine-Grained Openings for Decentralized Environments*. PhD thesis, Universidad Politécnica de Madrid, 2024.

KPT23. M. Kohlweiss, M. Pancholi, and A. Takahashi. How to compile polynomial IOP into simulation-extractable SNARKs: A modular approach. In *TCC 2023: 21st Theory of Cryptography Conference, Part III*, *Lecture Notes in Computer Science* 14371, pages 486–512, Taipei, Taiwan, November 29 – December 2, 2023. Springer, Cham, Switzerland.

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

KZ22. D. Kales and G. Zaverucha. Efficient lifting for shorter zero-knowledge proofs and post-quantum signatures. Cryptology ePrint Archive, Report 2022/588, 2022.

KZG10. A. Kate, G. M. Zaverucha, and I. Goldberg. Constant-size commitments to polynomials and their applications. In *Advances in Cryptology – ASIACRYPT 2010*, *Lecture Notes in Computer Science* 6477, pages 177–194, Singapore, December 5–9, 2010. Springer, Berlin, Heidelberg, Germany.

LPS24. H. Lipmaa, R. Parisella, and J. Siim. On knowledge-soundness of plonk in ROM from falsifiable assumptions. Cryptology ePrint Archive, Paper 2024/994, 2024.

LTZ24. X. Liu, I. Tzannetos, and V. Zikas. Adaptor signatures: New security definition and a generic construction for NP relations. In *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part II*, *Lecture Notes in Computer Science* 15485, pages 168–193. Springer, 2024.

Poe17. A. Poelstra. Scriptless scripts, 2017. `https://download.wpsoftware.net/bitcoin/wizardry/mw-slides/2017-03-mit-bitcoin-expo/slides.pdf`.

RRR16. O. Reingold, G. N. Rothblum, and R. D. Rothblum. Constant-round interactive proofs for delegating computation. In *48th Annual ACM Symposium on Theory of Computing*, pages 49–62, Cambridge, MA, USA, June 18–21, 2016. ACM Press.

Set20. S. Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In *Advances in Cryptology – CRYPTO 2020, Part III*, *Lecture Notes in Computer Science* 12172, pages 704–737, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Cham, Switzerland.

Sha79. A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

Ste06. J. Stern. A new paradigm for public key identification. *IEEE Trans. Inf. Theor.*, 42(6-P1):17571768, sep 2006.

ZLH+22. G. Zeng, J. Lai, Z. Huang, Y. Wang, and Z. Zheng. DAG-Σ: A DAG-based sigma protocol for relations in CNF. In *Advances in Cryptology – ASIACRYPT 2022, Part II*, *Lecture Notes in Computer Science* 13792, pages 340–370, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.

A Composition of Multi-Round Protocols via CDS

A.1 Monotone Access Structure

First we recall the definition of the monotone access structure from [CDS94].

Definition 11 (Monotone Access Structure [CDS94]). *An access structure $\Gamma \subset 2^M$ defined over a set M is called a monotone access structure if for all $A \in \Gamma$ and for all $B \supset A$ it holds that $B \in \Gamma$. Sets in Γ are called authorized sets, and sets not in Γ are called unauthorized sets.*

Definition 12 (Dual Structure [CDS94]). *Let Γ be an access structure defined over a set M. If $A \subseteq M$, then $\bar{A}$ denotes the complement of A in M. Now Γ^*, the dual access structure is defined as follows:*

$$A \in \Gamma^* \Leftrightarrow \bar{A} \notin \Gamma.$$

The dual Γ^* of a monotone access structure is also monotone, and satisfies $(\Gamma^*)^* = \Gamma$.
We also recall the following proposition from [CDS94].

Proposition 5 ([CDS94]). *Let Γ be monotone. A set is qualified in Γ if and only if it has a non-empty intersection with every qualified set in Γ^*.*

A.2 Secret Sharing Scheme

A semi-smooth perfect secret sharing scheme [CDS94], SSS_Γ, over domain S and access structure Γ over a set M consists of four polynomial-time algorithms, Share, Rec, $\mathsf{CheckShares}$ and $\mathsf{Complete}$ that:

$\mathsf{Share}_\Gamma(s) \to \{s^j\}_{j\in M}$: is a probabilistic algorithm that takes secret $s \in S$ and outputs shares $\{s^j\}_{j\in M}$. Let $D(s)$ the distribution of outputs from $\mathsf{Share}_\Gamma(s)$.

$\mathsf{Rec}_\Gamma(\{s^j\}_{j\in A}) \to s$: is a reconstruction algorithm that takes a qualified set of shares $\{s^j\}_{j\in A}$, $A \in \Gamma$, and recovers secret $s \in S$.

$\mathsf{CheckShares}_\Gamma(s, \{s^j\}_{j\in M})$: is a share verification algorithm that takes secret s and all shares $\{s^j\}_{j\in M}$, returns 1 or 0.

$\mathsf{Complete}_\Gamma(s, \{s^j\}_{j\in A})$: takes shares of a non-qualified set of shares $\{s^j\}_{j\in A}$ for $A \notin \Gamma$ and secret s, and outputs $\{s^j\}_{j\in \bar{A}}$ that $\{s^j\}_{j\in M}$ constitute a complete set of shares of s.

It provides the following properties:

Correctness. For all $s \in S$, $\{s^j\}_{j\in M} \leftarrow \mathsf{Share}_\Gamma(s)$, and $A \in \Gamma$, it holds that $s \leftarrow \mathsf{Rec}(\{s^j\}_{j\in A})$.

Perfect hiding. For any $A \notin \Gamma$, $s \in S$, and $\{s^j\}_{j\in M} \leftarrow \mathsf{Share}_\Gamma(s)$, the distribution of $\{s^j\}_{j\in A}$, denoted by D_A, is independent of s.

Consistency testing. $\mathsf{CheckShares}_\Gamma(s, \{s^j\}_{j\in M})$ returns 1 if and only if, for all $A \in \Gamma$, $\mathsf{Rec}_\Gamma(\{s^j\}_{j\in A}) = s$.

Share completion. For any $A \notin \Gamma$, $s \in S$, $\{s^j\}_{j\in A} \leftarrow\!\!\$\ D_A$, and $\{s^j\}_{j\in \bar{A}} \leftarrow \mathsf{Complete}_\Gamma(s, \{s^j\}_{j\in A})$, it holds that $1 \leftarrow \mathsf{CheckShares}_\Gamma(s, \{s^j\}_{j\in M})$ and $\{s_j\}_{j\in M}$ are distributed according to $D(s)$.

Efficient SSS_Γ exists for Γ being a threshold structure [Sha79], monotone circuit [BL90], and monotone span program [CDM00]. If, for every non-qualified set $A \notin \Gamma$, D_A equals uniform distribution over the corresponding share domain, then it is called a *smooth* perfect secret sharing scheme. Shamir's secret sharing scheme for threshold structures is an example [Sha79].

A.3 Insecurity of Naïve Multi-Round CDS

We describe CDS composition [CDS94] in a general form for $2\mu + 1$-move protocols. It matches the original composition when $\mu = 1$. We warn that the protocol is not sound for $\mu \geq 2$ and $k(> 3)$-special sound as mentioned earlier, and we only introduce it to establish notation and illustrate the problematic of the naïve composition.

Let Π^j be a $(2\mu+1)$-move public-coin proof protocol for relation R^j with a tuple of three algorithms $(\mathsf{A}^j, \mathsf{Z}^j, \mathsf{V}^j)$ (Definition 1), and (x^j, w^j) be a pair of instance and witness satisfying $R^j(x^j, w^j) = 1$. Let Γ be a monotone access structure over $[n]$, and $R_\Gamma((x^1, \ldots, x^n), (w^1, \ldots, w^n))$ be a compound relation that returns 1 if and only if there exists $A \in \Gamma$ that $R^j(x^j, w^j) = 1$ for all $j \in A$. We denote the special honest verifier zero-knowledge simulator of Π^j by Sim^j. Given Π^j for $j \in [n]$, an access structure Γ, and a semi-smooth secret sharing scheme $\mathsf{SSS}_{i,\Gamma^*}$ over a secret domain S_i and a share domain C_i for $i \in [\mu]$, a naïve CDS composition constructs Prover and $\mathsf{Verifier}$ as follows where steps $2i$ and $2i + 1$ are repeated for $i \in [\mu]$:

$\Pi^{\mathsf{cds}'}_{\mu,\Gamma}(\mathsf{Prover}(\{x^j\}_{j\in[n]}, \{w^j\}_{j\in A}), \mathsf{Verifier}(\{x^j\}_{j\in[n]}))$:

Step 1. For each $i \in [\mu]$, Prover samples $\{c_i^j\}_{j\in\bar{A}} \leftarrow D_{\bar{A}}$. Then Prover calls the special honest verifier zero-knowledge simulator $(a^j, \{z_i^j\}_{i\in[\mu]}) \leftarrow \mathsf{Sim}^j(x^j, \{c_i^j\}_{i\in[\mu]})$. For those $j \in A$, Prover commits to a^j by running $a^j \leftarrow \mathsf{A}^j(x^j; r^j)$. Prover sends $\{a^j\}_{j\in[n]}$ to $\mathsf{Verifier}$.

Step $2i$. $\mathsf{Verifier}$ samples $c_i \xleftarrow{\$} S_i$, and sends it to Prover.

Step $2i + 1$. Prover completes shares by $\{c_i^j\}_{j\in[n]} \leftarrow \mathsf{Complete}_{i,\Gamma^*}(c_i, \{c_i^j\}_{j\in\bar{A}})$, and computes $z_i^j \leftarrow \mathsf{Z}^j(x^j, w^j, \{c_m^j\}_{m\in[i]}; r^j)$ for all $j \in A$. It then sends $\{c_i^j, z_i^j\}_{j\in[n]}$ to $\mathsf{Verifier}$.

Step final. $\mathsf{Verifier}$ runs $\mathsf{V}^j(x^j, a^j, \{c_i^j\}_{i\in[\mu]}, \{z_i^j\}_{i\in[\mu]})$ for $j \in [n]$ and $\mathsf{CheckShares}_{i,\Gamma^*}(c_i, \{c_i^j\}_{j\in[n]})$ for $i \in [\mu]$, and outputs 1 if all outputs are 1, outputs 0, otherwise.

It is shown in [CDS94] that, if every Π^j is a 3-move public-coin proof protocol that is 2-special sound and honest verifier zero-knowledge, and Γ admits a smooth perfect secret sharing scheme, then the above protocol $\Pi^{\mathsf{cds}'}_{1,\Gamma}$ is a Σ-protocol for relation R_Γ. If every Π^j is honest verifier zero-knowledge, the above can be augmented to accept an access structure Γ that admits a smooth secret sharing scheme. It is done by modifying the prover's first step in a way that it first runs an HVZK simulator to directly obtain challenge c_i^j for $j \in \bar{A}$.

When $\mu \geq 2$, however, $\Pi^{\mathsf{cds}'}_{\mu,\Gamma}$ is not sound anymore. Consider a simple disjunctive access structure $\Gamma = \{\{A\}, \{B\}, \{A, B\}\}$. A cheating prover who knows witnesses for *both* A and B could strategically decide, round by round, whether to fix the challenge for the A-part or the B-part. For example, it might fix c_1^A and c_2^B. When an extractor tries to rewind this prover to get a full tree of transcripts (required for multi-round special soundness), it immediately fails. Rewinding on challenge c_1 would never change c_1^A (it was fixed by the prover), and rewinding on c_2 would never change c_2^B. Consequently, a complete extraction tree for either A or B cannot be formed, breaking the soundness of the protocol.

B Composition of Multi-Round Protocols via Share-then-Hash

To support the composition of an arbitrary polynomial number of instances, we generalize the Share-then-Hash method proposed in [AAB+20] with the help of a collision resistant hash (CRH) $H : \{0,1\}^* \mapsto \mathcal{C}_{i^*}$. Intuitively, rather than straightly use sampled values s^j as challenges $c_{i^*}^j = s^j$ for simulated transcripts, the prover is now required to obtain challenges $c_{i^*}^j \leftarrow H(\Gamma, \vec{x}, j, s^j)$ from the hash function.

The incorporation of CRH into protocol design introduces an inherent risk of extraction failure due to the adversary's potential successful guess and the hash collisions. Conversely, the security loss associated with perfect composition - previously exponential in nature - can be substantially reduced to a polynomial factor. To accommodate this probabilistic limitation while maintaining security guarantees, we adapt the *predicate special soundness* framework proposed in [AAB+24] and recall its relation to Fiat-Shamir knowledge soundness. Our composition with Share-then-Hash is presented in Figure 3. The security of the protocol is stated in Theorem 4.

B.1 Extending Predicate Special Soundness

We first recall the notations from [AAB+24].

Definition 13 (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, [AAB+24] introduced *challenge predicates* for a set of challenges. Intuitively, their challenge predicate checks whether a subset of challenges defining branches in the same level satisfies desired properties for extracting witness. In slightly more detail, the ℓ-th challenge predicate on each level takes the first $\ell - 1$ challenges $c_1, \ldots, c_{\ell-1}$, and checks that the current challenge c_ℓ falls in a "good" domain of challenges determined by the first $\ell - 1$ challenges. While this definition was sufficient for analyzing lattice-based proof systems in [AAB+24], it turns out that the analysis of Share-then-Hash demands challenge predicates with a different format of inputs. Looking ahead, a set of good challenges for Share-then-Hash is determined by the first $\ell - 1$ *responses* $z_1, \ldots, z_{\ell-1}$ following the challenges $c_1, \ldots, c_{\ell-1}$. We therefore generalize the notion of challenge predicates in such a way that they take *subtrees* $t_1, \ldots, t_{\ell-1}$ as input, encompassing

response-dependent good challenge domains.[15] Although the analysis of Share-then-Hash does not require entire subrees, we provide a more general form of definition.[16] Furthermore, while [AAB+24] required every predicate to be efficiently computable, their knowledge extractor for Fiat-Shamir-transformed protocols [AAB+24, Appendix H, Extractor 1] in fact only needs the ability to check whether *a chain of level m predicates $\Phi^{\mathsf{chal}}_{m,1},\ldots,\Phi^{\mathsf{chal}}_{m,k_m}$* simultaneously output 1 on a fixed set of subtrees. Therefore, we limit the computability requirement to a chain of predicates, rather than assuming every predicate is efficiently checkable. The definition below is of independent interest and may enable the analysis of other multi-round protocols with more complex extraction criteria.

Definition 14 (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 exits 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 $\Phi^{\mathsf{chal}}_{m,\ell}(t_1,\ldots,t_{\ell-1},c_\ell) = 0$.*

Definition 15 (Failure Density of Generalized Challenge Predicates). *Let $m \in [\mu]$ and $\ell \in [k_m]$. Define a set of good subtrees*

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

using the shorthand $c_i = \mathsf{chal}_m(t_i)$. 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 $(t_1,\ldots,t_{\ell-1}) \in \mathsf{GoodTree}_{m,\ell-1}$.

In the following, we adapt the predicate system of [AAB+24] with generalized challenge predicates, while omitting commitment predicates.

Definition 16 (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 17 (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 $R(x,w) = 1$.*

Adapting Theorem 5.1 of [AAB+24], we immediately obtain the following theorem on Fiat-Shamir knowledge soundness.

[15] We remark the idea here is similar in spirit to the *useful challenge structure* of Attema et al. [AKLY24], where a set of useful challenges are determined with respect to a sequence of transcripts that have been found.

[16] It is more general in the sense that one can always define a predicate that examines part of the subtree relevent to determination of good challenges.

Theorem 6 ([AAB+24] (adapted)). *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 3. While our generalized challenge predicate now takes subtrees as inputs instead of individual challenges, this modification does not affect the knowledge error analysis of [AAB+24]. This is because their extractor [AAB+24, Appendix H, Extractor 1]—which is itself adapted from [AFK22]—always obtains the first $\ell - 1$ subtrees (each containing one challenge in the current level) in a depth-first manner before sampling ℓ-th challenge. Thus, the extractor's failure probability can be conditioned on the existing $\ell - 1$ subtrees rather than on $\ell - 1$ individual challenges.

B.2 Analysis of Share-then-Hash in the Predicate Special Soundness Framework

To see why we need generalized challenge predicates, we first sketch the special soundness proof of Share-then-Hash. As in the proof of multi-round CDS, we set $k'_i = k_i$ for $i > i^*$ and $k'_i = n(k_i - 1) + 1$ for $i < i^*$ while parameterizing the number of critical round branches by $k'_{i^*} = \tau \geq n(k_{i^*} - 2) + 2$. Now, consider the set $\mathbb{T}_{i^*}$ of possible $(1, \ldots, 1, k'_{i^*}, \ldots, k'_\mu)$-trees for the protocol $\Pi^{\mathsf{sth}}_\Gamma$. Fix a subtree $t = (t_1, \ldots, t_\tau) \in \mathbb{T}_{i^*}$, where the i^*-th level challenge-response of each t_ℓ is denoted by $(s_\ell, \{z^j_\ell, s^j_\ell\}_{j\in[n]})$. (Note we omit the subscript i^* here.) Essentially, for witness extraction to succeed, we require that (1) $\mathsf{CheckShares}_{\Gamma^*}(s_\ell, \{s^j_\ell\}_{j\in[n]}) = 1$ for all $\ell = 1, \ldots, \tau$, and (2) for all $B \in \Gamma^*$, there exists $j \in B$ such that $|\{s^j_1, \ldots, s^j_\tau\}| \geq k_{i^*}$. (1) is already satisfied for any accepting transcript. Once (2) is satisfied, we can then rely on Proposition 5 to show the existence of a qualified set $A \in \Gamma$ by setting $A = \bigcup_{B\in\Gamma^*}\{j \in B : |\{s^j_1, \ldots, s^j_\tau\}| \geq k_{i^*}\}$. This implies that for each $j \in A$, one can find at least k_{i^*} distinct shares out of τ branches. Assuming collision-resistance of the hash function H, we can then conclude that there are at least k_{i^*} distinct challenges for each $j \in A$. In what follows, we formally define $\Phi^{\mathsf{chal}}_{i^*,1}, \ldots, \Phi^{\mathsf{chal}}_{i^*,\tau}$ such that (2) is satisfied if they all output 1.

Generalized Challenge Predicates and Failure Densities Let us denote a set $S^j_\ell = \{s^j_1, \ldots, s^j_\ell\}$ of j-th shares for given subtrees $(t_1, \ldots, t_\ell) \in \mathbb{T}^{(\ell)}_{i^*+1}$, and $\mathsf{Rec}_{\Gamma^*}(\{S^j_\ell\}_{j\in B})$ as a shorthand for $\{\mathsf{Rec}_{\Gamma^*}(\{s^j\}_{j\in B})\}_{s^j\in S^j_\ell}$. Intuitively, this denotes a set of secrets reconstructed from every possible combination of $|B|$ shares chosen from $\prod_{j\in B} S^j_\ell$. It is then easy to see that $|\mathsf{Rec}_{\Gamma^*}(\{S^j_\ell\}_{j\in B})| = \prod_{j\in B} |S^j_\ell|$. We now define the following generalized challenge predicates:

- For $\ell = 1$, $\Phi^{\mathsf{chal}}_{i^*,\ell}(s_\ell)$ returns 1 for any $s_\ell \in S$
- For $1 < \ell \leq k'_{i^*}$, where $k'_{i^*} = \tau \geq n(k_{i^*} - 2) + 2$, $\Phi^{\mathsf{chal}}_{i^*,\ell}(t_1, \ldots, t_{\ell-1}, s_\ell)$ returns 1 if the following conditions are satisfied and returns 0 otherwise:

$$\forall B \in \Gamma^* : \left(\exists j \in B : |S^j_{\ell-1}| \geq k_{i^*}\right) \vee s_\ell \notin \mathsf{Rec}_{\Gamma^*}(\{S^j_{\ell-1}\}_{j\in B}) \tag{2}$$

Before proving predicate special soundness, we prove the following utility lemma. By Lemma 1, it is also easy to see that a chain of the above predicates is efficiently checkable. That is, given $(t_1, \ldots, t_\tau) \in \mathbb{T}_{i^*}$, one can efficiently check if there exists a failing predicate by checking $\{j \in [n] : |S^j_\tau| \geq k_{i^*}\} \in \Gamma$.

Lemma 1. *Let $(t_1, \ldots, t_\tau) \in \mathbb{T}_{i^*}$ be a $(1, \ldots, 1, \tau, k_{i^*+1}, \ldots, k_\mu)$-tree of accepting transcripts and let $\tau \geq n(k_{i^*} - 2) + 2$. If $\Phi^{\mathsf{chal}}_{m,\ell}(t_1, \ldots, t_{\ell-1}, s_\ell) = 1$ for all $\ell \in [\tau]$, where s_ℓ is the i^*-th round challenge of t_ℓ, then there exists a qualified set $A \in \Gamma$ such that for all $j \in A$, $|S^j_\tau| \geq k_{i^*}$.*

Proof. Let $\{s^j_\ell\}_{j\in[n]}$ be a set of shares that reconstructs to the critical-round challenge s_ℓ in t_ℓ. Since each t_ℓ is accepting, we have that $\mathsf{CheckShares}_{\Gamma^*}(s_\ell, \{s^j_i\}_{j\in[n]}) = 1$ for every $\ell \in [\tau]$. We show that, for

every $B \in \Gamma^*$, there exists $j \in B$ such that $|S_\tau^j| \geq k_{i^*}$. If the predicate $\Phi_{i^*,\ell}^{\mathsf{chal}}$ outputs 1 by satisfying the first condition of (2) for some ℓ, then this statement is trivially true. Suppose the predicate outputs 1 by satisfying the second condition of (2) for all $\ell \in [\tau]$. For any $\ell > 1$ and $B \in \Gamma^*$, we have that $s_\ell = \mathsf{Rec}_{\Gamma^*}(\{s_\ell^j\}_{j\in B})$ by the consistency testing property of SSS_{Γ^*}. If $s_\ell \notin \mathsf{Rec}_{\Gamma^*}(\{S_{\ell-1}^j\}_{j\in B})$, then it must be that $s_\ell^j \notin S_{\ell-1}^j$ for some $j \in B$. Thus we guarantee that the sum $\sum_{j\in B} |S_\ell^j|$ is monotonically increasing with respect to ℓ. Since $\sum_{j\in B} |S_1^j| = |B|$, we have that

$$\sum_{j\in B} |S_\tau^j| \geq |B| + \tau - 1.$$

Thus, by setting $|B| + \tau - 1 \geq |B|(k_{i^*} - 1) + 1$, we can guarantee that for any $B \in \Gamma^*$, there exists some $j \in B$ such that $|S_\tau^j| \geq k_{i^*}$. Since $|B| \leq n$, we can set $\tau \geq n(k_{i^*} - 2) + 2$.

Finally, let $A = \bigcup_{B\in\Gamma^*} J_B$, where J_B denotes a set of $j \in B$ such that $|S_\tau^j| \geq k_{i^*}$. As each J_B is guaranteed to be non-empty, we have that $A \cap B \neq \emptyset$ for every $B \in \Gamma^*$, which implies $A \in \Gamma$ by Proposition 5. □

Next, we find the failure density of the above predicates.

Lemma 2. *The level i^* challenge predicates $p_{i^*,1}^{\mathsf{chal}}, \ldots, p_{i^*,k'_{i^*}}^{\mathsf{chal}}$ have the following failure densities:*

$$p_{i^*,\ell}^{\mathsf{chal}} = \begin{cases} 0 & \text{if } \ell = 1 \\ \frac{2^n \cdot (k_{i^*}-1)^n}{|S|} & \text{if } 1 < \ell \leq k'_{i^*} \end{cases}$$

Proof. The case $\ell = 1$ is trivial. For $1 < \ell \leq k'_{i^*}$, we first fix $\ell - 1$ "good" subtrees $(t_1, \ldots, t_{\ell-1}) \in \mathsf{GoodTree}_{i^*,\ell-1}$. If $\Phi_{i^*,\ell-1}^{\mathsf{chal}}$ outputs 1 by meeting the first condition of (2) for some $B \in \Gamma^*$, then the subsequent $\Phi_{i^*,\ell}^{\mathsf{chal}}$ always outputs 1 for the same B regardless of s_ℓ. Thus, to find the uppder bound on the size of bad challenges, we assume that for all $B \in \Gamma^*$, $s_{\ell-1} \notin \mathsf{Rec}_{\Gamma^*}(\{S_{\ell-2}^j\}_{j\in B})$. By the definition of $\Phi_{i^*,\ell}^{\mathsf{chal}}$, we have that

$$\begin{aligned} &|\mathsf{BadChal}_{i^*,\ell}(t_1, \ldots, t_{\ell-1})| \\ &\leq \sum_{B\in\Gamma^*} \left| \left\{ s \in S \,|\, (\forall j \in B : |S_{\ell-1}^j| \leq k_{i^*} - 1) \wedge s \in \mathsf{Rec}_{\Gamma^*}(\{S_{\ell-1}^j\}_{j\in B}) \right\} \right| \\ &\leq \sum_{B\in\Gamma^*} (k_{i^*} - 1)^{|B|} \leq 2^n \cdot (k_{i^*} - 1)^n. \end{aligned}$$

Thus, we obtain the failure density $p_{i^*,\ell}^{\mathsf{chal}} = 2^n \cdot (k_{i^*} - 1)^n / |S|$. □

Analysis of Predicate Special Soundness We are now ready to prove our main result, Theorem 4.

Proof. **Completeness.** It directly follows from the completeness and critical-round special honest verifier zero-knowledge of Π^j. We note that $\mathsf{Complete}_{\Gamma^*}$ works for the set $\bar{A}$ of shares $\{s^j\}_{j\in\bar{A}}$ since $\bar{A} \notin \Gamma^*$ when $A \in \Gamma$.

Critical-Round Special Honest Verifier Zero-Knowledge. Consider the following simulator for $\Pi_\Gamma^{\mathsf{sth}}$ that proceeds as follows, given as input $\{x^j\}_{j\in[n]} \in \mathcal{L}_{R_\Gamma}$, $\{c_i\}_{i\neq i^*}$ and $s \in S$:

1. Compute $\{s^j\}_{j\in[n]} \leftarrow \mathsf{Share}_{\Gamma^*}(s)$
2. For all $j \in [n]$: $c_{i^*}^j = H(\Gamma, \vec{x}, j, s^j)$ and $(st_1^j, a^j) \leftarrow \mathsf{SimA}^j(x^j, c_{i^*}^j)$
3. For all $i \in [\mu]$ and $j \in [n]$: $(st_{i+1}^j, z_i^j) \leftarrow \mathsf{SimZ}^j(st_i^j, c_i^j)$, where $c_i^j = c_i$ if $i \neq i^*$
4. Output $(\{a^j\}_{j\in[n]}, \{c_i\}_{i\in[\mu]\setminus\{i^*\}}, \{z_i^j\}_{i\in[\mu],j\in[n]}, \{s^j\}_{j\in[n]}, s)$

The perfect hiding and share completion properties of SSS_{Γ^*} imply that the distribution of $\{s^j\}_{j\in[n]}$ generated as above is identical to that of a real transcript. Since $\{x^j\}_{j\in[n]} \in \mathcal{L}_{R_\Gamma}$, for $j \in \bar{A}$, the simulator generates a^j, z_i^j exactly as a real prover; for $j \in A$, the distribution of a^j, z_i^j output by CRSHVZK simulators for Π^j is indistinguishable with the one generated by a real prover. Grouping Step 1-2 together as SimA and Step 3 as SimZ, respectively, the above algorithm is indeed a valid CRSHVZK simulator.

Predicate Special Soundness. The proof can be viewed as generalization of [CDS94]. However, we must carefully choose the special soundness parameters for $\Pi_{\Gamma}^{\mathsf{sth}}$ in such a way that the corresponding tree of accepting transcripts contains a well-structured tree for every $j \in A \in \Gamma$ from which the extractor Ext^j for Π^j can successfully extract witness for x^j. We construct an efficient extractor $\mathsf{Ext}^{\mathsf{sth}}$ that determines some $A \in \Gamma$ and then internally runs sub-extractors $\{\mathsf{Ext}^j\}_{j\in A}$ to output a valid set of witnesses $\mathbf{w} = \{w^j\}_{j\in A}$, given as input any $\vec{k}'$-tree T^{sth} of accepting transcripts.

We first recall the structure of the tree T^{sth} for $\Pi_{\Gamma}^{\mathsf{sth}}$. Each path in T^{sth} defines a transcript

$$(\{a^j\}_{j\in[n]}, \{c_i\}_{i\in[\mu]\setminus\{i^*\}}, \{z_i^j\}_{i\in[\mu],j\in[n]}, s, \{s^j\}_{j\in[n]})$$

which can be decomposed into n individual transcripts:

$$\{(a^j, \{c_i\}_{i\in[\mu]\setminus\{i^*\}}, c_{i^*}^j, \{z_i^j\}_{i\in[\mu]})\}_{j\in[n]}$$

where $c_i^j = c_i$ for $i \neq i^*$ and $c_{i^*}^j = H(\Gamma, \vec{x}, j, s^j)$ for $j \in [n]$. Thus, T^{sth} can also be decomposed into n individual trees, $T^1, \ldots, T^n$, where every path of T^j is derived from decomposition of the corresponding path in T^{sth}. We first observe the following properties:

1. In non-critical rounds $i \neq i^*$, the challenges in the decomposed trees T^j for $j \in [n]$ remain the same as the corresponding challenges in T^{sth}.
2. In round i^*, the shares s^j in the decomposed trees T^j for $j \in [n]$ pass the consistency check performed by $\mathsf{CheckShares}_{\Gamma^*}$ with respect to the corresponding secret s in T^{sth}.

The analyses of Case 1 ($i^* < i \leq \mu$) and Case 3 ($i < i^*$) are identical to those of Theorem 3.

To prove Case 2 ($i = i^*$), we show the following statement: if $k'_{i^*} = \tau \geq n(k_{i^*} - 2) + 2$, then for any $(1, \ldots, 1, k'_{i^*}, k_{i^*+1}, \ldots, k_\mu)$-subtree $T_{i^*}^{\mathsf{sth}} = (t_1, \ldots, t_{k'_{i^*}})$ for $\Pi_{\Gamma}^{\mathsf{sth}}$ such that $\Phi_{i^*,\ell}^{\mathsf{chal}}(t_1, \ldots, t_{\ell-1}, s_\ell) = 1$ for $\ell = 1, \ldots, k'_{i^*}$, there exists $A \in \Gamma$ such that for every $j \in A$, the corresponding decomposed subtree $T_{i^*}^j$ contains a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-subtree $\tilde{T}_{i^*}^j$ for statement x^j. By Lemma 1, we have that there exists $A \in \Gamma$ such that for all $j \in A$, the corresponding shares $s_1^j, \ldots, s_{k'_{i^*}}^j$ are pairwise distinct. Thus, assuming that the outputs $c_{i^*,\ell}^j$ of the hash function $H(\Gamma, \vec{x}, j, s_\ell^j)$ never collide for distinct inputs, for each $j \in A \in \Gamma$, one can derive from $T_{i^*}^j$ a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-subtree $\tilde{T}_{i^*}^j$ of accepting transcripts for statement x^j.

Extractor. Combining Case 1-3 above, there exists $A \in \Gamma$ such that for every $j \in A$, the corresponding decomposed tree T^j contains a $\vec{k}$-tree $\tilde{T}^j$ for Π^j. The extractor $\mathsf{Ext}^{\mathsf{sth}}$ then simply runs sub-extractors Ext^j on $\tilde{T}^j$ to extract witnesses for all $j \in A$. As Π^j is $\vec{k}$-SS, every Ext^j always succeeds in extracting witness.

□

C Other Definitions

Definition 18 (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.*

For more notation related to the $(k_1, \ldots, k_\mu)$-tree of transcripts T. Outside the notations introduced in Section 3.2, some extra notations are introduced below to support other definitions. By $\mathsf{nonleaves}(T)$, we denote all node indices of T except for the leaves. Let $\mathsf{nodes}(T, i)$ denote the set of node indices in depth i of T, e.g., $\mathsf{nodes}(T, 2) = \{(1, 1), (1, 2), \ldots, (1, k_1)\}$. For every $\mathcal{C}_i$ and $k_i > 1$, we denote sets of pairwise distinct k_i challenges by

$$\mathcal{C}_{k_i}^{\mathsf{dis}} := \{(c_1, \ldots, c_{k_i}) \,|\, \forall j, \ell(\neq j) \in [k_i], (c_j, c_\ell) \in \mathcal{C}_i \times \mathcal{C}_i, c_j \neq c_\ell\}.$$

Let $\mathsf{pnum}(T, i)$ denote the number of parent nodes at level i, i.e., $\mathsf{pnum}(T, i) = \prod_{j=1}^{i} k_{j-1}$ for $k_0 = 1$.

Definition 19 (Statistical $(k_1, \ldots, k_\mu)$-Special Soundness [ABO+24]). *A $(2\mu+1)$-round public-coin proof protocol is statistical $(k_1, \ldots, k_\mu)$-special sound with knowledge error ϵ_{stss} if there exists a polynomial-time algorithm* Ext *such that for any adversary $\mathcal{A}$, it holds that:*

$$\Pr\left[\begin{array}{ll} & T \text{ is a } (k_1,\ldots,k_\mu)\text{-tree} \\ & \text{of accepting transcripts} \\ \wedge & \left(\forall i \in [\mu-1], \forall id \in \overrightarrow{[k_i]}, \right. \\ & \left. \mathsf{chal}(T, id) = \{c_{id||j}\}_{j\in[k_{i+1}]}\right) \\ \wedge & R(x,w) = 0 \end{array} \begin{array}{l} \\ \forall i \in [\mu], \forall id \in \overrightarrow{[k_i]}, c_{id} \leftarrow\!\$\ \mathcal{C}_i \\ : (x, T) \leftarrow \mathcal{A}(\{c_{id}\}_{id\in\overrightarrow{[k_\mu]}}) \\ w \leftarrow \mathsf{Ext}(x, T) \\ \\ \end{array}\right] \leq \epsilon_{\mathsf{stss}},$$

where $\overrightarrow{[k_i]} = 1 \times [k_1] \times \cdots \times [k_i]$. A tree of transcript is called "bad" if the extractor fails.

It is stressed that error bound ϵ_{stss} is independent of messages from the prover. It is shown in [ABO+24] that, a parallel repetition of $(k_1, \ldots, k_\mu)$-special sound protocol results in a statistical $(k_1, \ldots, k_\mu)$-special sound protocol with negligible statistical soundness error. They also showed that it constitutes a proof of knowledge system.

Definition 20 (Round-by-Round Soundness [CCH+19,FGQ+23]). *Let $\Pi = (\mathsf{A}, \mathsf{Z}, \mathsf{V})$ be $(2\mu+1)$-move public-coin proof protocol. We say that Π is round-by-round sound if, there exists a "doomed set" $\mathcal{D} \in \{0,1\}^*$ such that,*

- *If $x \notin \mathcal{L}$, then $(x, \varnothing) \in \mathcal{D}$, where $\varnothing$ denotes the empty transcript.*
- *For all (2ℓ)-move partial transcript $\tau = (a, c_1, z_1, \ldots, z_{\ell-1}, c_\ell)$, such that $(x, \tau) \in \mathcal{D}$, for all next message z_ℓ given by the prover, there exists a negligible function $\mathsf{negl}(\cdot)$ such that*

$$\Pr_{c_{\ell+1}\leftarrow\mathcal{C}_{\ell+1}}[(x, \tau, z_\ell, c_{\ell+1}) \notin \mathcal{D}] \leq \mathsf{negl}(\lambda).$$

- *For any x, any (2μ)-move partial transcript τ and any last prover message z_μ, if $(x, \tau) \in \mathcal{D}$ then $V(x, \tau, z_\mu) = 0$.*

A Σ-protocol [Cra96] is a three-move public-coin proof protocol that is 2-special sound and special honest verifier zero-knowledge. The 2-special soundness implies optimal soundness defined as follows.

Definition 21 (Optimal Soundness [FHJ20]). *A three-move public-coin proof protocol for relation R is optimally sound if, for any $x \notin \mathcal{L}_R$, and $a \in \{0,1\}^*$, there exists at most one challenge $c \in \mathcal{C}$ that there exists z that $\mathsf{V}(x, a, c, z) = 1$ holds.*

Definition 22 (Predicate Special Soundness [AAB+24]). *Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a $2\mu+1$-message public-coin argument of knowledge for a relation R_{PP}. We say that Π is $(\mathbf{K}, \mathbf{\Phi})$-predicate-special sound for $\mathbf{K} = (\mathbf{k_1}, \cdots, \mathbf{k}_\mu)$ and a predicate system $\mathbf{\Phi}$ if there exists a polynomial time algorithm which given a statement x and a $\mathbf{K}$-tree of transcripts for this statement with $\mathbf{\Phi}(\mathbf{t}) = \mathbf{1}$ always outputs a witness w such that $w \in R_{\mathsf{PP}}(x)$.*

Definition 23 ($\mathfrak{G}$-soundness [DFMS22]). *Let $\mathfrak{G} \subseteq 2^C$ be increasing. For a non-empty $\mathfrak{G}$, a $(2\ell+1)$-move identification protocol Π is called $\mathfrak{G}$-sound if there exists a probabilistic polynomial time algorithm $\mathsf{Ext}_{\mathfrak{G}}$ that takes as input*

- *a public key* pk *generated by* Keygen*, and*
- *a set $\mathcal{T}$ of transcripts whose*
  - *first message are the same, that is, $\forall \mathbf{t}, \mathbf{t}' \in \mathcal{T}$, $\mathbf{t}_{<1} = \mathbf{t}'_{<1}$,*
  - *challenge sequences $\mathbf{c}(\mathbf{t})$, $\mathbf{t} \in \mathcal{T}$ form a set $\{\mathbf{c}(\mathbf{t}), \mathbf{t} \in \mathcal{T}\} \in \mathfrak{G}$,*
  - *transcripts pass verification, that is, $\forall \hat{\mathbf{t}} \in \mathcal{T}$, $\mathsf{Vrt}_\Pi(\mathsf{pk}, \hat{\mathbf{t}}) = 1$,*

  and outputs a secret key sk *such that* $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{Keygen}$*. We say $\mathfrak{G}$ is an extraction structure for Π.*

Definition 24 (Special Unsoundness [AFK23,BGTZ23]). *Let $\Pi = (\mathcal{P}, \mathcal{V})$ be $(2\mu+1)$-move public-coin proof protocol, and let $(\ell_1, \ldots, \ell_\mu) \in \mathbb{N}^\mu$. We say that Π has $(\ell_1, \ldots, \ell_\mu)$-special unsoundness if there exists a dishonest prover $\mathcal{A}$ of the following form and, so that in the execution with $\mathcal{V}$ and input x the following holds:*

– *$\mathcal{A}$ starts off in active mode, which is so that in every round i, when $\mathcal{A}$ sends the message m_i, there exists a subset $\mathcal{L}_i \subseteq \mathcal{C}_i$ such that $|\mathcal{L}_i| = \ell_i$ (defined as a function of the state of $\mathcal{A}$ at that point) such that if the subsequent challenge c_i is in $\mathcal{L}_i$, then $\mathcal{A}$ switches into passive mode.*
– *If $\mathcal{A}$ switches into passive mode, then it remains in passive mode until the end of the protocol, and $\mathcal{V}$ accepts at the end of the protocol.*

Definition 25 (k-Zero Knowledge [GKK+22,DG23]). *Let $\Pi = (P, V)$ be a $(2\mu + 1)$-move public-coin interactive proof protocol with HVZK simulator Sim, and $k \in [\mu]$. Let Π_{FS} be its associated FS-transformed NIZK. We say Π_{FS} satisfies (perfect) k-zero knowledge (k-ZK) if there exists a zero-knowledge simulator $\mathsf{Sim}_{\mathsf{FS},k}$ that only needs to program the random oracle in round k, and whose output is identically distributed to that of honestly generated proofs.*

Following the literature [AABN02,GOP+22,DG23], we exploit the concept of min-entropy to represent the likelihood of the first message a of an interactive protocol taking on some fixed value.

Definition 26 (Min-entropy of first message). *Let λ be a security parameter and language $\mathcal{L}_R$ be an NP language associated by R. Consider a pair $(x, w) \in R$ and let $\Pi = (P, V)$ be a multi-round interactive proof. Let $\mathrm{Coin}(\lambda)$ be the set of coins used by the prover and $\mathsf{A}(\cdot)$ the algorithm that outputs the first message from P. Consider the set $A(x, w) = \{\mathsf{A}(x, w; r) : r \leftarrow\!\$\ \mathrm{Coin}(\lambda)\}$, i.e. the set of all possible first round messages associated to (x, w). The min-entropy function associated to Π is defined as*

$$\zeta(\lambda) = \min_{(x,w)} \left(-\log_2 \eta(x, w)\right),$$

where the minimum is taken over all possible (x, w) drawn from R, and $\eta(x, w)$ is the maximum probability that a first message takes on a particular value, i.e.,

$$\eta(x, w) = \max_{a \in A(x,w)} (\Pr\left[\mathsf{A}(x, w; r) = a \mid r \leftarrow\!\$\ \mathrm{Coin}(\lambda)\right]).$$

We say $\Pi = (P, V)$ has sufficient min-entropy if $\zeta(\lambda)$ is super-logarithmic in λ.

C.1 Relations to Other Notions

In this section, we compare CRZK with related notions, including its reduction to k-zero knowledge and its implication of k-special unsoundness.

CRZK leads to k-zero knowledge. We recap the notion of k-zero-knowledge (Definition 25) that was first introduced in [GKK+22] and formalized in [DG23]. Informally, an interactive proof is k-zero-knowledge if there exists a zero-knowledge simulator $\mathsf{Sim}_{\mathsf{FS}}$ that only needs to program the random oracle in round k, and whose transcript is indistinguishable from honestly generated ones, stated as follows:

Theorem 7. *Let Π be a $(2\mu+1)$-move public-coin proof protocol which is critical-round special honest verifier zero-knowledge at round i^*_{zk} and has sufficient min-entropy (Definition 26). Let Π_{FS} be the corresponding non-interactive proof protocol via Fiat-Shamir transform. Then Π_{FS} is k-zero knowledge with $k = k_{i^*_{\mathsf{zk}}}$.*

Proof. (Sketch) We construct a simulator, $\mathsf{Sim}_{\mathsf{FS}}$, that fulfills requirements in [DG23], as follows:

1. Given statement x as input, sample $c_{i^*_{\mathsf{zk}}} \leftarrow\!\$\ \mathcal{C}_{i^*_{\mathsf{zk}}}$ and obtain $(st_1, a) \leftarrow \mathsf{SimA}(x, c_{i^*_{\mathsf{zk}}})$.
2. Query the random oracle and get $c_i = H(x, a, z_1, \ldots, z_{i-1})$ for all $i \neq i^*_{\mathsf{zk}}$. Specially if $i = 1$, get $c_1 = H(x, a)$.
   If $i = i^*_{\mathsf{zk}}$, reprogram $H(x, a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1}) := c_{i^*_{\mathsf{zk}}}$.
   $\mathsf{Sim}_{\mathsf{FS}}$ aborts if $H(x, a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1})$ has already been defined.
3. Obtain $(st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)$.

Since Π has sufficient min-entropy, the probability that $\mathsf{Sim}_{\mathsf{FS}}$ aborts in Step 2 is negligible. The distribution of $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ is indistinguishable from that of honestly generated transcripts due to the critical-round special honest verifier zero-knowledge property. □

k-CRZK and k-special unsoundness. Our k-CRZK implies k-special unsoundness [AFK23,BGTZ23] at the critical round i^*_{zk} for languages that constitute a hard subset membership problem [GW11]. Special unsoundness (Definition 24) claims the ability of a cheating prover to convince the verifier of a false statement if the challenge sent by the verifier is a bad challenge at certain round. We can build the cheating prover by running the k-CRZK simulator on $k-1$ preselected challenges. If the verifier "unluckily" sends one of these $k-1$ challenges at the critical round, the cheating prover is able to behave honestly in the remaining rounds. Due to the hardness of the subset membership problem with respect to the concerned language, the CRZK simulator works on false statements as required. The reverse implication does not generally hold since special unsoundness does not concern the output distribution of the cheating prover.

D Critical Rounds in the KKW Framework

D.1 KKW Framework

We succinctly describe 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.

Next, we describe the resulting MPCitH protocol. 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.

D.2 Critical Round in KKW

Theorem 8. *Given the 5-move interactive honest-verifier zero-knowledge proof in [KKW18] (denoted as KKW), assuming that the hash function used is collision-resistant and the commitment scheme used is computationally binding and hiding, then KKW is flexible and can be:*

- *Critical-round special honest-verifier zero-knowledge at round $i^*_{\mathsf{zk}} = 1$ or at round $i^*_{\mathsf{zk}} = 2$.*
- *$\vec{k}$-critical-round special sound where $i^*_{\mathsf{ss}} = 1,\ \vec{k} = (2, 1)$ or where $i^*_{\mathsf{ss}} = 2,\ \vec{k} = (1, 2)$.*

Proof. We prove the flexibility property of KKW with respect to CRSS and critical-round HVZK. Specifically, we prove that KKW satisfies CRSS for both $i^*_{\mathsf{ss}} = 1$ and $i^*_{\mathsf{ss}} = 2$ by first constructing an extractor ExExt to show that KKW is extended $(2, 2)$-special sound. From ExExt, we then construct an extractor CrExt that, given access to a valid proof transcript, extracts a valid witness (not the solution of R_{hd}) while ensuring that the tree of transcripts is defined according to the CRSS definition. For critical-round HVZK, we construct a pair of simulators $(\mathsf{SimA}, \mathsf{SimZ})$ that simulate the real execution in both cases, thereby satisfying the required indistinguishability conditions. We denote a transcript of KKW protocol by (a, c_1, z_1, c_2, z_2).

CRSS.
Given an extended $(2, 2)$-tree of transcripts T with four accepting transcripts that belong to T, our goal is to show that there exists an extended extractor ExExt that can either extract an actual witness, or find a collision on the commitment scheme or the hash function. For each case, we construct ExExt as below.
Case 1: $i^*_{\mathsf{ss}} = 1, \vec{k} = (2, 1)$. To see this, we note that the prefix (a, c_1, z_1) where $c_1 \in [m]$ commits to all m executions of the preprocessing phase $\{\mathsf{seed}_j\}_{j\in[m]}$ and then opens all of them except for the c_1-th. Given a transcript (a, c_1, z_1, c_2, z_2), we now bulid an extractor ExExt as follows:

- From the transcript $(a, c'_1, z'_1, *, *)$ of the tree where $c_1 \neq c'_1$, it allows ExExt to get c_1-th values of preprocessing seed_{c_1}. From seed_{c_1}, ExExt computes $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$.
- We note that z_1 includes the masked values $\{\hat{z}_\alpha\}$ for the *input wires* that is computed from w and $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$. From $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$ and then together with $\{\hat{z}_\alpha\}$, ExExt effectively computes a witness w.

We show that $C(w) = 1$, i.e., ExExt successes in extracting the witness or finds a collision in commitments and hash function. Consider another transcript $(a, c_1, z_1, c'_2, z'_2)$ having the same prefix (a, c_1, z_1) and $c_2 \neq c'_2$.

- From $c_2 \neq c'_2$ and (z_2, z'_2), ExExt learns all $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$ along with $(com_{c_1,c_2}, \mathsf{msgs}_{c_2}, com_{c_1,c'_2}, \mathsf{msgs}_{c'_2})$.
- We highlight that as h is fixed in round 1, and as $\{\hat{z}_\alpha\}, h'$ (the commitment of $\{\mathsf{msgs}_i\}_{i\in[n]}$) are fixed in round 3, then all $\{\mathsf{state}_{c_1,i}, \mathsf{msgs}_i\}_{i\in[n]}$ obtained in the two accepted transcripts are consistent with an honest execution of MPC protocol unless there is a break in the binding of the commitment scheme or the collision resistance of H. And in that case, ExExt outputs the collision as a solution to R_{hd}.

Case 2: $i^*_{\mathsf{ss}} = 2, \vec{k} = (1, 2)$. Given an accepted transcript (a, c_1, z_1, c_2, z_2), the extractor ExExt works as follows:

- From z_1, ExExt learns $\{\mathsf{seed}_j\}_{j \neq c_1}$, h' and $\{\hat{z}_\alpha\}$.
- Consider another transcript $(a, c_1, z_1, c'_2, z'_2)$ with the same prefix (a, c_1, z_1) and $c_2 \neq c'_2$ since $c_2 \neq c'_2$ then from z'_2, ExExt learns all $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$. We note that, ExExt also learns $(com_{c_1,c_2}, \mathsf{msgs}_{c_2})$ because they are included inside of z_2.
- From $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$, ExExt learns the value λ_α as the masks of each input wire, therefore from $\{\hat{z}_\alpha\}$, ExExt can effectively compute a witness w.

As in the first case, we argue that $C(w) = 1$. To prove this, we need to show that each $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$ are generated honestly from a seed seed_{c_1} obtained during the preprocessing phase. ExExt does the following:

- Consider another accepting transcript $(a, c'_1, z'_1, *, *)$ of T where $c_1 \neq c'_1$. As $c_1 \neq c'_1$, the extractor ExExt can recover seed_{c_1} from z'_1, as this value was generated honestly in the preprocessing phase.
- Given seed_{c_1}, ExExt computes $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$, which must be consistent with the values outputted in the fifth move, z_2. 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 $a := h$.

Constructing CrExt. In both cases, CrExt is implicit in the construction of ExExt. In fact, conditioned on successful ExExt *not* outputting a solution to R_{hd}, from any $(2, 1)$-subtree or $(1, 2)$-subtree of an extended tree T, one can extract witness without any failure. Observe that, given a tree T, the extractor ExExt only requires two transcripts,

- $(a, c_1, z_1, *, *)$ and $(a, c'_1, z'_1, *, *)$ as in the first case $i^*_{\mathsf{ss}} = 1$,
- or (a, c_1, z_1, c_2, z_2) and $(a, c_1, z_1, c'_2, z'_2)$ as in the second case $i^*_{\mathsf{ss}} = 2$,

to extract a unique value w. The probability that ExExt successfully extracts w is 1. Moreover, this extracted w is indeed a valid witness; otherwise, using another accepted transcript, ExExt would output either a collision of the hash function H or two distinct inputs breaking the binding property of the commitment scheme.

Therefore, given a deterministic tree T, the extractor CrExt is constructed in the same way as ExExt, stopping as soon as it obtains w. Since ExExt succeeds with T, it follows that w is a valid witness.

Critical-round HVZK.

Case 1: $i^*_{\mathsf{zk}} = 1$. The critical-round special HVZK at $i^*_{\mathsf{zk}} = 1$ is proven by constructing two simulators $(\mathsf{SimA}(x, c_1), \mathsf{SimZ})$ as follows,

- $\mathsf{SimA}(x, c_1) \to (st_1, a)$: For a known first-round challenge c_1, the SimA does as follows:
  For all $j \in [m]$, SimA generates honestly as an honest prover to get all m preprocessing materials including $\mathsf{seed}_j, \{\mathsf{seed}_{j,i}\}_{i\in[n]}, \mathsf{aux}_{j,n}$, defines $\mathsf{state}_{j,i} := \mathsf{seed}_{j,i}$ for $i \in [n-1]$ and $\mathsf{state}_{j,n} := \mathsf{seed}_{j,n} \| \mathsf{aux}_{j,n}$.
  For $j = c_1$, SimA samples uniformly $\{\hat{z}_\alpha\}$ as masked value of input wire of circuit C. Together with $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$, SimA executes the MPC protocol until the reconstruction output step. At this step Sim learns the shares of output wire $[\![b]\!]_i$ of all i-th party ($i \in [n]$), Sim then defines an auxiliary value aux_b such that $[\![b]\!]_i \oplus \mathsf{aux}_b = 1$.
  SimA redefines the $\mathsf{aux}_{c_1,n}$ as before, except that the auxiliary value corresponding to the last multiplication gate of C is now defined by XORing with aux_b and $\mathsf{state}_{c_1,n}$ is redefined as $\mathsf{seed}_{c_1,n} \| \mathsf{aux}_{c_1,n}$ using new value of $\mathsf{aux}_{c_1,n}$.
  Next, the SimA computes each value h_j for $j \in [m]$ as the honest prover would. Finally, the SimA computes $h = H(h_1, \ldots, h_m)$ honestly. Output $st_1 := (\{\mathsf{state}_{j,i}\}_{j\in[m],i\in[n]}, \{\hat{z}_\alpha\})$ and $a := h$.
- $\mathsf{SimZ}(st_1, c_1) \to (z_1, st_2)$. To simulate the third message, SimZ starts by retrieving the states from st_1. Then, it executes *honestly* online phase of MPC protocol to get the broadcast messages $\{\mathsf{msgs}_i\}_{i\in[n]}$ using $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$. Finally, it computes $h' = H(\mathsf{msgs}_1, \ldots, \mathsf{msgs}_n)$ from the state. Output $st_2 := (st_1, \{\mathsf{msgs}_i\}_{i\in[n]})$ and $z_1 := (\{\mathsf{seed}_j\}_{j\neq c_1}, h', \{\hat{z}_\alpha\})$.

– $\mathsf{SimZ}(st_2, c_2) \to z_2$: To simulate the fifth message, from st_2, simply output, as the honest prover would do.
Output $z_2 := (\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, com_{c_1,c_2}, \mathsf{msgs}_{c_2})$.

Correctness. We first show that the transcript $\{\mathsf{seed}_{c_1,i}\}_{i\in[n]}$, $\{\hat{z}_\alpha\}$, $\mathsf{aux}_{c_1,n}$ produced by the simulator allows the MPC protocol to output 1. We focus on the last multiplication gate of circuit C, where random shares are assigned to this gate. Denote the shares of its input wires as $[\![\lambda_\alpha]\!]$ and $[\![\lambda_\beta]\!]$, while the shares of the output wire are $[\![\lambda_{\alpha\cdot\beta}]\!]$. For $i \in [n-1]$, the i-th party generates $[\![\lambda_{\alpha\cdot\beta}]\!]_i$ from the random seed $\{\mathsf{seed}_{c_1,i}\}$, and the n-th party uses $\mathsf{aux}_{c_1,n}$ to define its share. By the way the auxiliary value is defined for this gate in SimA, the n-th party gets $[\![\lambda_{\alpha\cdot\beta}]\!]_n \oplus \mathsf{aux}_b$. Therefore, after all parties broadcast their shares and reconstruct them, they obtain a value $\hat{z}_\gamma = z_\gamma \oplus \lambda_\gamma \oplus \mathsf{aux}_b$ (λ_γ is the masked value for this wire).

The online phase continues until all parties reconstruct the circuit's output. We note that after executing the last multiplication gate to obtain $\hat{z}_\gamma$, $\mathsf{aux}_{c_1,n}$ is no longer used, and there are no broadcast messages from the parties until the final step of reconstructing the mask of the output value b, since the $\{\hat{z}_\alpha\}$ of addition gates can be computed locally. Therefore, aux_b is xored into all $\{\hat{z}_\alpha\}$ of the gates that is after the last multiplication gate.

For the step of reconstructing the output of the circuit, every party outputs $[\![b]\!]_i \oplus \mathsf{aux}_b = 1$.

Indistinguishability. We then prove that the transcripts produced by the simulators (a, c_1, z_1, c_2, z_2) are computationally indistinguishable from those generated during actual protocol executions with an honest verifier.
We observe that $z_1 := (\{\mathsf{seed}_j\}_{j\neq c_1}, h', \{\hat{z}_\alpha\})$ is indistinguishable from the real protocol since $\{\mathsf{seed}_j\}_{j\neq c_1}, \hat{z}_\alpha$ are sampled uniformly as in the real protocol, and h' is computed by executing the MPC protocol *honestly* from seed_{c_1} and $\mathsf{aux}_{c_1,n}$. We note that although SimA emulates $\mathsf{aux}_{c_1,n}$, in this round, the verifier does not have any information about seed_{c_1} or $\mathsf{aux}_{c_1,n}$.
Turning to $z_2 := (\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, com_{c_1,c_2}, \mathsf{msgs}_{c_2})$, we first note that com_{c_1,c_2} is computed exactly according to the protocol description, ensuring consistency with the initial message $a := h$. We now argue that even though SimA emulates $\mathsf{aux}_{c_1,n}$, the values $(\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, \mathsf{msgs}_{c_2})$ remain indistinguishable from their real values in the actual protocol.

– If $c_2 = n$, then $\{\mathsf{state}_{c_1,i}\}_{i\neq n}$ are indeed sampled randomly as in the honest execution.
– If $c_2 \neq n$, the verifier knows $\mathsf{state}_{c_1,n} = \mathsf{seed}_{c_1,n} \| \mathsf{aux}_{c_1,n}$ but does not know seed_{c_1,c_2}. Consequently, the verifier cannot verify whether $\mathsf{aux}_{c_1,n}$ was honestly generated by checking the correctness of the masked value shares for the final multiplication gates.

Moreover, the simulated values ensure that the MPC output is 1, and all broadcast messages $\{\mathsf{msgs}_i\}_{i\in[n]}$ are computed honestly following MPC protocol's description. Therefore, $(\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, \mathsf{msgs}_{c_2})$ remain indistinguishable from the real values.
Case 2: $i^*_{\mathsf{zk}} = 2$.

The CRSHVZK property at $i^*_{\mathsf{zk}} = 2$ is proven by constructing two simulators $(\mathsf{SimA}(x, c_2), \mathsf{SimZ})$. We build these simulators based on the semi-honest security of the MPC protocol, for which there exists a simulator Sim that outputs simulated consistent views of $n-1$ parties, i.e., all except for the c_2-th party, as follows. Sim starts by sampling $\{\mathsf{state}_i\}_{i\neq c_2}, \{\hat{z}_\alpha\}, \mathsf{msgs}_{c_2}$ and uses these sampled values to simulate the online phase of the MPC protocol until the reconstruction output step. At this step Sim learns the shares of output $[\![b]\!]_i$ of i-th party ($i \neq c_2$), Sim then defines a $[\![b]\!]_p$ such that $\oplus[\![b]\!]_i = 1$, appends $[\![b]\!]_p$ into msgs_{c_2} and gets a new broadcast message msgs_{c_2}. Therefore, we can modify Sim syntactically by redefining its input to be a set of $\{\mathsf{state}_i\}_{i\neq c_2}, \{\hat{z}_\alpha\}$ that are sampled randomly in advance, such that $\mathsf{Sim}(\{\mathsf{state}_i\}_{i\neq c_2}, \{\hat{z}_\alpha\}) \to \mathsf{msgs}_{c_2}$.

Now, we define the simulators $(\mathsf{SimA}, \mathsf{SimZ})$ as follows, for challenges $c_1 \leftarrow\!\$\ [m], c_2 \leftarrow\!\$\ [n]$.

– $\mathsf{SimA}(x, c_2) \to (st_1, a)$: For a known second-round challenge c_2, the simulator does as follows. First, SimA chooses uniform seeds as the honest prover would do, obtains seed_j for $j \in [m]$ and from seed_j computes honesly $\{\mathsf{seed}_{j,i}\}_{i\in[n]}$. Then, for each $j \in [m]$, SimA computes $\mathsf{aux}_{j,n}$ from $\{\mathsf{seed}_{j,i}\}_{i\in[n-1]}$. SimA defines $\mathsf{state}_{j,i} = \mathsf{seed}_{j,i}$ for all $i \in [n-1]$ and $\mathsf{state}_{j,n} := \mathsf{seed}_{j,n} \| \mathsf{aux}_{j,n}$, and samples $\{\hat{z}_{j,\alpha}\}_{j\in[m]}$ honestly.
Then, SimA runs the MPC simulator $\mathsf{Sim}(\{\mathsf{state}_{j,i}\}_{j\in[m],i\neq c_2}, \{\hat{z}_{j,\alpha}\}_{j\in[m]}) \to \{\mathsf{msgs}_{j,c_2}\}_{j\in[m]}$. This is done m times to simulate m independent preprocessing phases. Next, the SimA computes each

value h_j for $j \in [m]$ as the honest prover would. Finally, the SimA computes $h = H(h_1, \ldots, h_m)$ honestly.
Output $st_1 := (\{\mathsf{state}_{j,i}\}_{j\in[m], i\neq c_2}, \{\hat{z}_{j,\alpha}\}_{j\in[m]}, \{\mathsf{msgs}_{j,c_2}\}_{j\in[m]})$ and $a := h$.

- $\mathsf{SimZ}(st_1, c_1) \to (z_1, st_2)$: To simulate the third message, SimZ starts by retrieving the states from st_1. Then, it computes the remaining $\mathsf{msgs}_{c_1,i}$ honestly for every $i \neq c_2$. Finally, it computes $h' = H(\mathsf{msgs}_{c_1,1}, \ldots, \mathsf{msgs}_{c_1,n})$ from the state.
  Output $st_2 := \{st_1, \mathsf{msgs}_{c_1,c_2}\}$ and $z_2 := (\{\mathsf{seed}_j\}_{j\neq c_1}, h', \{\hat{z}_{c_1,\alpha}\})$.
- $\mathsf{SimZ}(st_2, c_2) \to z_2$: To simulate the fifth message, simply output, as the honest prover would do, the MPC simulation corresponding to the c_1-th preprocessing phase except for the view of c_2-th party.
  Output $z_2 := (\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, com_{c_1,c_2}, \mathsf{msgs}_{c_1,c_2})$.

Due to the security of the MPC protocol against all-but-one corruption, it is easy to see that by a standard hybrid argument, the transcripts produced by the simulator (a, c_1, z_1, c_2, z_2) are computationally indistinguishable from those generated during actual protocol executions with an honest verifier.

□

E More Applications

E.1 Accumulators

An accumulator, also known as a set commitment, is a succinct primitive that allows one to commit to a set of elements $S = \{x_1, \ldots, x_t\}$ and then produce a short proof of membership for any $x_i \in S$ [Bd94,BP97]. We introduce a definition below, following the syntax in [Kol24].

Definition 27 (Accumulator). *A (static) accumulator with domain $\mathcal{M}$ is a tuple of algorithms $(\mathsf{Gen}, \mathsf{Accum}, \mathsf{WitGen}, \mathsf{Ver})$ with the following syntax:*

$\mathsf{Gen}(1^\lambda) \to ck$: *On input the security parameter λ and an upper bound for the set size n, $\mathsf{Gen}(1^\lambda)$ returns a commitment key ck.*

$\mathsf{Accum}(ck, S) \to (com, st)$: *On input a commitment key ck and a set $S \subset \mathcal{M}$, $\mathsf{Accum}(ck, S)$ outputs a commitment com and a state st.*

$\mathsf{WitGen}(ck, st, m) \to \pi$: *On input a commitment key ck, a state st and a value m, $\mathsf{WitGen}(ck, st, m)$ outputs a membership proof π.*

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

Moreover, these algorithms should satisfy the following properties:

Completeness. *For any $S \subset \mathcal{M}$ such that $|S| \leq n$ and for any $m \in S$,*

$$\Pr\left[\mathsf{Ver}(ck, com, m, \pi) = 1 \;\middle|\; \begin{array}{l} ck \leftarrow \mathsf{Gen}(1^\lambda, n) \\ (com, st) \leftarrow \mathsf{Com}(ck, S) \\ \pi \leftarrow \mathsf{WitGen}(ck, st, m) \end{array}\right] = 1.$$

Soundness. *For any PPT adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{l} \mathsf{Ver}(ck, com, m^*, \pi^*) = 1 \\ \wedge\, m^* \notin S^* \end{array} \;\middle|\; \begin{array}{l} ck \leftarrow \mathsf{Gen}(1^\lambda, n) \\ S^* \leftarrow \mathcal{A}(ck) \\ (com, st) \leftarrow \mathsf{Com}(ck, S^*) \\ (m^*, \pi^*) \leftarrow \mathcal{A}(ck, com, st) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Succinctness. *Both com and π have size bounded by $\mathcal{O}(\lambda, \mathsf{polylog}(n))$. The running time of Ver is also bounded by $\mathcal{O}(\lambda, \mathsf{polylog}(n))$.*

To construct our accumulators, we describe a slightly stronger variation of CRZK which requires a simulator that takes $k-1$ possible challenges for the critical round. We introduce it below.

Definition 28 (Critical-Round k-Special Honest Verifier Zero-Knowledge). *A $(2\mu+1)$-move public-coin proof protocol is critical-round k-special honest verifier zero-knowledge (k-CRSHVZK or simply k-CRZK for short) at round i^*_{zk} if there exists a set of polynomial-time algorithm* $(\mathsf{SimAX}, \mathsf{SimZX})$ *that:*

- SimAX *takes x and $(c^{(1)}_{i^*_{\mathsf{zk}}}, \ldots, c^{(k-1)}_{i^*_{\mathsf{zk}}}) \in \mathcal{C}^{k-1}_{i^*_{\mathsf{zk}}}$ as input, and outputs (st_1, a) where st_1 is a state information and a is a prover's message at step 1.*
- SimZX *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 $(x, w) \in L_{RW}$, $c_i \in \mathcal{C}_i$ for $i \in [\mu] \setminus \{i^*_{\mathsf{zk}}\}$, $(c^{(1)}_{i^*_{\mathsf{zk}}}, \ldots, c^{(k-1)}_{i^*_{\mathsf{zk}}}) \in \mathcal{C}^{k-1}_{i^*_{\mathsf{zk}}}$, and any $c_{i^*_{\mathsf{zk}}} \in (c^{(1)}_{i^*_{\mathsf{zk}}}, \ldots, c^{(k-1)}_{i^*_{\mathsf{zk}}})$, distribution of $(a, c_1, z_1, \ldots, c_\mu, z_\mu)$ generated as $(st_1, a) \leftarrow \mathsf{SimAX}(x, c^{(1)}_{i^*_{\mathsf{zk}}}, \ldots, c^{(k-1)}_{i^*_{\mathsf{zk}}})$, $(st_{i+1}, z_i) \leftarrow \mathsf{SimZX}(st_i, c_i)$ 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.*

Construction of an Accumulator. We generalize our construction of a trapdoor commitment scheme (Figure 4) to construct an accumulator $\mathsf{ACC}_{\Pi_{i^*}}$ from a $(2\mu+1)$-move public-coin proof protocol Π_{i^*} that is critical-round k_{i^*}-special honest verifier zero knowledge and critical-round special sound at round $i^*_{\mathsf{ss}} = i^*_{\mathsf{zk}} = i^*$ and where $k_{i^*} \geq 3$. Our accumulator scheme requires that the maximum size of n that is supported by $\mathsf{Gen}(1^\lambda, n)$ is bounded by the parameter $B = k_{i^*} - 1$ that depends on Π_{i^*}. We present our construction in Figure 6.

Recall that critical-round k_{i^*}-special HVZK (Definition 28) requires that there exists a zero-knowledge simulator $(st_1, a) \leftarrow \mathsf{SimZX}(x, c^{(1)}_{i^*}, \ldots, c^{(k-1)}_{i^*})$ where each $c^{(j)}_{i^*} \in \mathcal{C}_{i^*}$, i.e., that takes $k_{i^*} - 1$ challenges to simulate the first message of the protocol a.[17] A peculiar consequence is that the message space $\mathcal{M} \subset \mathcal{C}_{i^*}$ must be a proper subset such that $|\mathcal{M}| \leq |\mathcal{C}_{i^*}| - B$. This is necessary for the proof of soundness to go through. The reason is that a commitment to S always needs to be simulated based on B challenges, and if $|S| < B$, the challenges not in S cannot be part of the message space to guarantee that the extraction of a complete tree of transcripts is successful. Hence, we need to "taint" B values from $\mathcal{C}_{i^*}$, which will be used in the simulation but cannot be part of the message space.

Theorem 9. *The construction $\mathsf{ACC}_{\Pi_{i^*}}$ in Figure 6 constitutes an accumulator if Π_{i^*} is a perfectly complete and critical-round k_{i^*}-special honest verifier zero-knowledge interactive proof at round $i^*_{\mathsf{zk}} = i^*$. It is binding if G and H are non-programmable and programmable random oracles, respectively, relation R is one-way, and Π_{i^*} is $\vec{k}$-critical-round special sound at round $i^*_{\mathsf{ss}} = i^*$ for which $k_{i^*} \geq 3$.*

Proof. One can see by inspection that the construction satisfies correctness if $n \leq B$. The construction also satisfies succinctness as both *com* and π are independent of n and therefore of size $\mathcal{O}(\lambda)$.

For soundness, let $\mathcal{A}$ be an adversary against accumulator soundness with non-negligible success probability. We will show how to use $\mathcal{A}$ as a black box to construct an adversary $\mathcal{B}$ that breaks the one-wayness of the relation R. For this, it suffices to show that $\mathcal{B}$ can use $\mathcal{A}$ to obtain a valid $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-subtree of transcripts T^* contained in the extended special soundness tree T. The proof follows a similar strategy as the proof of Theorem 5, so we skip some technical details.

The reduction $\mathcal{B}$ is given a challenge instance x and access to a non-programmable random oracle G. $\mathcal{B}$ also simulates the programmable random oracle H for $\mathcal{A}$. As in the previous proof, $\mathcal{B}$ forwards all queries of $\mathcal{A}$ to G, behaving transparently, and follows a lazy (uniformly random) simulation strategy for H. $\mathcal{B}$ sets up the soundness game for $\mathcal{A}$ by setting $ck = x$. Then, $\mathcal{A}(ck)$ outputs a set $S = \{m_1, \ldots, m_t\} \subset \mathcal{M}$ such that $t \leq n$.

Next, $\mathcal{B}$ runs $(com, st) \leftarrow \mathsf{Com}(ck, S)$ where $com = (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$. Then, it runs $(m^*, \pi^*) \leftarrow \mathcal{A}(ck, com, st)$ which returns a valid proof π^* for a message $m^* \notin S$, which $\mathcal{B}$ parses as $\pi^* = (m^*, z^*_j, c^*_{j+1}, \ldots, c^*_\mu, z^*_\mu)$. Note that $m^* \in \{m_{t+1}, \ldots, m_{k_{i^*}-1}\}$ is not valid since $m_{t+1}, \ldots, m_{k_{i^*}-1}$ are out of the message space. Recall that the goal of $\mathcal{B}$ is to obtain a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$ tree of transcripts. For this, note that $\mathcal{B}$ can simulate $k_{i^*} - 1$ valid sub-trees that start at level i^* by simply running $(st_{i+1}, z_{i+1}) \leftarrow \mathsf{SimZX}(st_i, c_{i+1})$ for all the required branches and their corresponding challenges, where at level i^* it sets $c^{(j)}_{i^*} = m_j$ for every $j = 1, \ldots, k_{i^*} - 1$ (recall that the accumulator

[17] One example of such protocol is the three-move Stern Σ-protocol [Ste06], which is trivially 3-critical at its only round.

Set up.

- Let $\Pi_{i^*} = (\mathsf{A}, \mathsf{Z}, \mathsf{V}, \mathsf{SimAX}, \mathsf{SimZX})$ be a $(2\mu+1)$-move public-coin proof for relation R that is (1) k_{i^*}-CRSHVZK at round $i^*_{\mathsf{zk}} = i^*$, and (2) $\vec{k}$-CRSS at round $i^*_{\mathsf{ss}} = i^*$ where $k_{i^*} \geq 3$.
- Let $B = k_{i^*} - 1$ and $\mathcal{C}_{i^*}$ be the challenge space of Π_{i^*} at the i^*-th round.
- Let the message space $\mathcal{M} \subset \mathcal{C}_{i^*}$ be a proper subset such that $|\mathcal{M}| \leq |\mathcal{C}_{i^*}| - B$.
- Let G and H be two hash functions.

Construction.

Gen: Given 1^λ and a size bound n as input, check whether $n \leq B$ and otherwise abort. Sample an instance $(x, w) \leftarrow\!\!\$\ \mathcal{L}_{RW}(\lambda)$. The commitment key is $ck := x$.

Accum: Given (ck, S) as input, parse $ck = x$ and $S = (m_1, \ldots, m_t)$. Abort if $t > n$, otherwise do:

- If $t < B$, choose $B - t$ arbitrary elements $m_{t+1}, \ldots, m_B \in \mathcal{C}_{i^*} \setminus \mathcal{M}$.
- Compute $(st_1, a) \leftarrow \mathsf{SimAX}(x, m_1, \ldots, m_B)$.
- For $i = 1, \ldots, i^* - 1$, do

$$c_i := G(x, a, c_1, z_1, \ldots, c_{i-1}, z_{i-1}) \tag{3}$$

  and $(st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)$.
- For $i = i^*$ and for $j = 1, \ldots, t$, do $c_i^{(j)} := m_j$ and $(st_{i+1}^{(j)}, z_i^{(j)}) \leftarrow \mathsf{SimZX}(st_i, c_i^{(j)})$.
- For $i = i^* + 1, \ldots, \mu$ and $j = 1, \ldots, t$, do

$$c_i^{(j)} := H(x, a, c_1, z_1, \ldots, c_{i-1}^{(j)}, z_{i-1}^{(j)}) \tag{4}$$

  and $(st_{i+1}^{(j)}, z_i^{(j)}) \leftarrow \mathsf{SimZX}(st_i^{(j)}, c_i^{(j)})$.
- Output $com := (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$ and $st := \left\{ m_j, z_{i^*}^{(j)}, c_{i^*+1}^{(j)}, z_{i^*+1}^{(j)}, \ldots, c_\mu^{(j)}, z_\mu^{(j)} \right\}_{j=1,\ldots,t}$.

WitGen: Given (ck, m, st) as input, check whether $m = m_j$ for some $m_j \in st$, and otherwise abort. Then, parse st to output $\pi_j = (m_j, z_{i^*}^{(j)}, c_{i^*+1}^{(j)}, z_{i^*+1}^{(j)}, \ldots, c_\mu^{(j)}, z_\mu^{(j)})$.

Ver: Given (ck, com, m, π) as input, parse them into $(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$ with $c_{i^*} := m$. Check if every c_i satisfies the relations in (3) and (4), and output $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.

Fig. 6. Accumulator $\mathsf{ACC}_{\Pi_{i^*}}$.

st contains messages $m_1, \ldots, m_t, m_{t+1}, \ldots, m_{k_{i^*}-1}$, where $S = \{m_1, \ldots, m_t\}$ and $m_{t+1}, \ldots, m_{k_{i^*}-1} \notin \mathcal{M}$).

Therefore, to get a valid CRSS tree of transcripts T^*, $\mathcal{B}$ only needs one additional sub-tree that starts at level i^*. The missing sub-tree can be obtained by setting $c_{i^*}^{(k_{i^*})} = m^*$ and by rewinding $\mathcal{A}$ while reprogramming the random oracle H to provide distinct (uniformly random) challenges at every iteration, following the same steps as in the proof of Theorem 5.

It remains to show that the tree of transcripts satisfies the conditions of Definition 10. This again follows as in the previous proof, as it is obtained by running the adversary with uniformly chosen challenges. Then, from Theorem 1, we conclude that the CRSS extractor CrExt outputs a witness from T^* with non-negligible probability. Thus, once T^* is obtained, $\mathcal{B}$ runs $w \leftarrow \mathsf{CrExt}(T)$ and returns w. □

Trapdoor Commitment from $k_{i^*} \geq 2$. Inspired by our accumulator, we present a feasibility result to show how to build a trapdoor commitment scheme if Π_{i^*} is CRZK and CRSS with $k_{i^*} \geq 2$ at the critical round, generalizing the $k_{i^*} \leq 2$ result from earlier sections. The idea is to extend the openings of the trapdoor commitment scheme to include $t = \lceil k_{i^*}/2 \rceil$ transcripts (branching at level i^*) in the opening information. Then, any adversary who breaks commitment binding provides k_{i^*} transcripts branching at level i^*, and security follows via a rewinding argument as in the previous constructions.

We also need that all challenges at level i^* are distinct. For this, we use an encoding $\gamma : \mathcal{M} \to \mathcal{C}_{i^*}^t$ such that for any element $c \in \mathcal{C}_{i^*}$, there exists at most one element $m \in \mathcal{M}$ such that $c \in \gamma(m)$. Intuitively, this condition ensures that any challenge $c \in \mathsf{Im}(\gamma)$ is associated to a unique $m \in \mathcal{M}$. We describe the commit and verify algorithms in more detail:

$\mathsf{Com}(ck, m)$**:** Start by computing $(st_1, a) \leftarrow \mathsf{SimAX}(x, \gamma(m))$. Without loss of generality, we assume the simulator takes t challenges; it can just use elements of $\mathcal{C}_{i^*} \setminus \mathsf{Im}(\gamma)$ if more are required.

– For $i = 1, \ldots, i^* - 1$, compute c_i and z_i as in Figure 6.

– For $i = i^*$ and for $j = 1, \ldots, t$, do $c_i^{(j)} := \gamma(m)_j$. and $(st_{i+1}^{(j)}, z_i^{(j)}) \leftarrow \mathsf{SimZX}(st_i, c_i^{(j)})$.
– For all remaining $i = i^* + 1, \ldots, \mu$ and $j = 1, \ldots, t$, compute the challenges $c_i^{(j)}$ and messages $z_i^{(j)}$ as in Figure 6.
Output:

$$com := (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1}),$$
$$open := \left\{\gamma(m)_j, z_{i^*}^{(j)}, c_{i^*+1}^{(j)}, z_{i^*+1}^{(j)}, \ldots, c_\mu^{(j)}, z_\mu^{(j)}\right\}_{j=1,\ldots,t}.$$

$\mathsf{Ver}(ck, com, m, open)$**:** Check the validity of every c_i until round $i^* - 1$ as in Figure 6. Then, compute $c_{i^*}^{(j)} := \gamma(m)_j$ for $j = 1, \ldots, t$ and check the validity of every $c_i^{(j)}$ for $j = i^* + 1, \ldots, \mu$. Finally, check that
$\mathsf{V}(x, a, c_1, z_1, \ldots, c_i^{(j)}, z_i^{(j)}, \ldots, c_\mu^{(j)}, z_\mu^{(j)}) = 1$ for every $j = 1, \ldots, t$.

E.2 Trapdoor Commitments with Flexible Trapdoor Allocation

Observe that the MR-CDS composition in Section 5.2 is closed for some class of protocols, i.e., the resulting proof is also critical-round zero-knowledge. Thus, it can be seamlessly combined with our construction of the trapdoor commitment scheme in Section 6.

This modular construction allows for flexible trapdoor setup regarding a monotone access structure of one's interest. For instance, by combining proofs for statements A and B into $A \vee B$ using the composition, and using the composed critical-round proof system to build the commitment scheme, we obtain a trapdoor commitment scheme that is equivocal only if the committer knows a witness for either A or B. One could alternatively base the construction on a critical-round proof system for NP. However, this modular approach is particularly advantageous when the underlying relation can be decomposed into sub-relations that each admit efficient critical-round proofs, such as KKW protocols or standard sigma-protocols. This enables more efficient and flexible instantiations of the trapdoor commitment scheme, tailored to the structure of the specific relations involved.