Critical Round in Multi-Round Proofs: Compositions and Transformation to Trapdoor Commitments

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

[1] NTT Social Informatics Laboratories
abe.masayuki@iecl.ntt.co.jp
[2] IMDEA Software Institute,
Universidad Politécnica de Madrid,
NTT Social Informatics Laboratories
david.balbas@imdea.org
[3] IRIF, Université Paris Cité
bui@irif.fr
[4] NICT
m.ohkubo@nict.go.jp
[5] Kyoto University
shang.zehua.23m@st.kyoto-u.ac.jp
[6] NTT Social Informatics Laboratories
mehdi.tibouchi@ntt.com

Abstract. In many multi-round public-coin interactive proof systems, challenges in different rounds serve different roles, but a formulation that actively utilizes this aspect has not been studied extensively. In this paper, we propose new notions called *critical-round special honest verifier zero-knowledge* and *critical-round special soundness*. Our notions are simple, intuitive, easy to apply, and capture several practical multi-round proof protocols including, but not limited to, those from the MPC-in-the-Head paradigm.

We demonstrate the usefulness of these notions with two fundamental applications where three-round protocols are known to be useful, but multi-round ones generally fail. First, we show that critical-round proofs yield trapdoor commitment schemes. This result also enables the instantiation of post-quantum secure adaptor signatures and threshold ring signatures from MPCitH, resolving open questions in (Haque and Scafuro, PKC 2020) and in (Liu et al., ASIACRYPT 2024). Second, we show that critical-round proofs can be securely composed using the Cramer-Schoenmakers-Damgård method. This solves an open question posed by Abe et al. in CRYPTO 2024.

Overall, these results shed new light on the potential of multi-round proofs in both theoretical and practical cryptographic protocol design.

Keywords: Multi-Round, Critical Round, Composition, Trapdoor Commitment, MPCitH, Adaptor Signatures, Threshold Ring Signatures

1 Introduction

1.1 Background

A public-coin proof system is a widely used tool in cryptographic protocol design. In its simplest three-move form, the prover, attempting to prove a relation on instance x, first commits to a, receives a challenge c from the verifier, and responds with z. The verifier accepts if (x, a, c, z) satisfies the verification predicate. This framework not only leads to non-interactive zero-knowledge proof systems and signature schemes via the Fiat-Shamir transformation [FS87], but also has various other important applications. One simple yet surprising example are trapdoor commitment (TD) schemes [Dam90], which can be constructed from any Σ-protocol where x serves as the commitment key, a is the commitment, the message is embedded in c, and z is the opening information. If the committer knows the witness for x, they can open a to any message c by using z, as the proof protocol can be completed with any challenge. Conversely, if the witness is unknown, the committer can only

* This work was done while Dung Bui was doing a summer internship at NTT Social Informatics Laboratories.

respond to a single pre-determined challenge c through zero-knowledge simulation, leveraging the special honest verifier zero-knowledge property of Σ-protocols.

Recent advances in design paradigms have led to the development of efficient $(2\mu + 1)$-move public-coin proof systems. The MPC-in-the-Head (MPCitH) paradigm [IKOS07] demonstrates how to construct zero-knowledge proofs from a multi-party computation (MPC) protocol. Its preprocessing variants [KKW18,BN20] result in multi-round proofs[7] with concrete efficiency. This framework has also been applied extensively to the development of efficient post-quantum signatures, e.g., [BDK+21,FJR22,KZ22,AGH+23,CCJ23,BCC+24]. Another paradigm that produces multi-round proofs is interactive oracle proofs (IOPs) [RRR16,BCS16], as well as polynomial IOPs [BFS20]. These have given rise to a vast number of constructions of succinct non-interactive arguments of knowledge (SNARKs) such as [AHIV17,BCR+19,CHM+20,GWC19,COS20,CFF+21], and to polynomial commitment schemes [BBHR18,ACFY24].

The security of multi-round proofs has been studied primarily in terms of their soundness properties. Several formulations of soundness and their relationships are studied in the literature [BGTZ23]. With regard to knowledge soundness, $(k_1, \ldots, k_\mu)$-special soundness [ACK21,AFR23] is a natural generalization of the standard 2-special soundness [Cra96] used in three-move proofs. The formulation is primarily used to analyze the security of non-interactive proofs and signature schemes resulting from the Fiat-Shamir transformation. A variation of zero-knowledge is considered in [DG23,GKK+22] for analyzing the security of SNARKs via the Fiat-Shamir transformation in the programmable random oracle model.

Beyond non-interactive zero-knowledge proofs and signatures, further applications for multi-round proofs should be explored to expand their utility both in theory and practice. However, extending the rich results of three-move proofs to multi-round ones is far from trivial. Take, for instance, the aforementioned trapdoor commitments—where to embed the message among the multiple challenges is unclear. In fact, it can be argued that, due to the non-interactive nature of the commitment scheme, the transformation from a three-move proof to a trapdoor commitment schemes does not extend to the multi-round case in general. It is also noted in [ABO+24] that the multi-round analogue of the well-known Cramer-Shoenmakers-Damgård (CDS) composition [CDS94] does not yield sound proofs. We elaborate on these unsuccessful applications of multi-round proofs in the next section.

1.2 Our Contribution

Given that multi-round proof protocols often face challenges in adapting well-developed techniques for three-move ones, our objective is to introduce a useful formulation that characterizes a class of multi-round proof protocols that are as effective as their three-move counterparts. General multi-round public-coin protocols differ from the three-move protocols in two key ways: the number of moves $2\mu+1$, which exceeds three; and the number of branches k_i at each step, which can exceed two. In this paper, we primarily focus on the issues arising from the increased number of rounds. Whenever our results impose restrictions on k_i, we state this explicitly and discuss possible extensions to more general cases.

The results of this paper are summarized as follows:

1. **Utility-Focused Characterization.** Observing that challenges in multi-round proofs often serve different roles, we introduce the notion of a *critical round*, defined as the round that plays a critical role in both zero-knowledge and (knowledge) soundness. We formalize the notions of *critical-round special honest verifier zero-knowledge (CRZK)* and *critical-round special soundness (CRSS)*, which together characterize a class of multi-round proofs we call *critical-round protocols*. CRZK states that the whole proof can be simulated only by knowing a challenge in the specific round in advance, and aims for construction of applications. CRSS is a combination of round-by-round soundness and special soundness switched in the specific round. It helps to build more compact tree of transcripts or eliminating programmable random oracles.
   The class of critical-round protocols includes important existing constructions of multi-round proofs, such as those based on MPCitH and IOPs. To demonstrate this, we formally prove that the first MPCitH protocol with preprocessing, also known as the KKW protocol [KKW18], satisfies these notions.

[7] Some are in the three-move format with a structured challenge string. We view them as multi-round with separate challenges.

2. **Application 1: Trapdoor Commitment.** We present a transformation from critical-round proofs to trapdoor commitment schemes for $k = 2$ at their critical round. For $k > 2$, we instead construct an *accumulator* that binds $k - 1$ messages into a single string.
   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. Our construction enables to obtain them solely from (practically efficient) MPCitH, answering the open question posed in both [LTZ24] and [HS20].
3. **Application 2: Composition.** We show that critical-round protocols are securely composed via the CDS composition solely by applying it to the critical round with $k = 2$. This partially answers an open question posed in [ABO+24] that seeks for compositions as powerful as the CDS for multi-round protocols. We also argue that our composition extends to general $k > 2$ in the random oracle model applying the Share-then-Hash technique from [AAB+20]. As a further application, we introduce trapdoor commitments with flexible trapdoor allocation that combines this result with the first one.

To justify our focus on the above applications, we explain how general multi-round proof protocols fail in these cases. A key observation is that the zero-knowledge simulator is employed in the *construction*. Since the general form of the zero-knowledge simulator requires to input all challenges to generate a simulated transcript, the protocols can only be executed if all challenges for the simulated part are available in advance. This establishes a sharp contrast with non-interactive proofs via Fiat-Shamir, e.g., [AAB+24,HJMN24], where the zero-knowledge simulator is used only in the security proof with a programmable random oracle.

Failure in trapdoor commitment: Recall the trapdoor commitment scheme from a three-move protocol: the commitment key is an instance x where the witness is not known to the committer. To commit to message m, the committer runs the zero-knowledge simulator with m as challenge c and obtains a as a commitment and z as an opening information.
Consider a multi-round analogue of this construction. The committer would set all challenges $c_1, \ldots, c_\mu$ to be m (or compute them deterministically from m) and run the zero-knowledge simulator to get a and all $z_1, \ldots, z_\mu$. In the binding game, the adversary will produce *two* transcripts for the same a but with distinct messages m and m'. However, for the $(k_1, \ldots, k_\mu)$-special soundness extractor to work, we need a tree of transcripts that branches in all later challenges as well. It is clear that rewinding does not help at all since the game is non-interactive. One might consider resorting to the Fiat-Shamir approach where each $c_i = H(x, a, c_1, \ldots, z_{i-1})$ with $c_1 = m$. Modeling H as a programmable random oracle allows rewinding. However, the initial message a is available only *after* the zero-knowledge simulator has been executed. Thus, the honest committer cannot obtain any of the c_i in advance.

Failure in composition: Consider the CDS composition with a minimal example of the disjunctive statement $A \vee B$. A prover who only has a witness for A must simulate the proof for B using the zero-knowledge simulator. For this to be possible, all challenges for the B part must be prepared in advance. This means that, under the CDS paradigm, all real challenges provided by the verifier will be secret-shared between the A and B parts, such that the B challenges are set as prepared, while the A challenges follow the randomness of the real challenges. However, this is no longer sound if there are two or more rounds. Consider a cheating prover (who potentially knows both witnesses) preparing the first half of the rounds for the A challenges and the latter half of the rounds for the B challenges in advance, then following the protocol with these pre-determined challenges. Rewinding such a prover never yields a sufficient set of transcripts for extraction because half of the shared challenges are fixed for both A and B.

1.3 Related Work

On special soundness and special zero-knowledge. Special soundness is defined in [Cra96] as the existence of a knowledge extractor that, given *any* two accepting transcripts for the same initial message, outputs a witness *with probability one*. Based on an observation that parallel repetition of k-special sound protocols for $k > 2$ does not preserve k-special soundness, it is relaxed to screen

erroneous transcripts with a predicate in [AAB+21]. A multi-round generalization of special soundness, known as $(k_1, \ldots, k_\mu)$-special soundness, is introduced in [AF22,AFR23]. As well as it is observed in the three-move case, $(k_1, \ldots, k_\mu)$-special soundness does not capture a class of protocols that repeats atomic protocols in parallel [AF22]. It is then relaxed to handle erroneous transcripts in several ways [ABO+24,AAB+24,DFMS22].

The aim of special soundness is to separate the adversary from the knowledge extractor. Consequently, it becomes difficult to account for extraction errors based on computational assumptions regarding the adversary. This issue is ingeniously addressed in [AFR23] by extending the extractor to output, with probability 1, either a witness for the relation or a witness for the computational problem. In this manner, it is possible to work with arguments such as KKW, which rely on the hardness of a hash function collision.

The notion of k-zero knowledge in [DG23] shares essentially the same idea as our critical-round zero-knowledge. They formalize it in the programmable random oracle model with a perfect zero-knowledge flavor, as required for proving the Fiat-Shamir security of SNARKs. We extend and refine this concept for interactive protocols, which is crucial for broadening its applicability.

On round elimination paradigm. Round elimination is a powerful technique developed in [HJMN24] that transforms a class of multi-round protocols into three-move ones in the random oracle model. In our terminology, it applies to protocols where the last round is critical, such that applying the Fiat-Shamir transformation up to the last round leaves only the final round as interactive. This technique is useful for constructing efficient non-interactive zero-knowledge proofs, allowing for a modular security analysis. However, it does not guarantee a black-box use of the resulting three-move protocol, as the zero-knowledge property holds only in the programmable random oracle model. Because of this limitation, it is unsuitable to first apply round elimination to obtain a three-move protocol and then use it for CDS composition or trapdoor commitments.

On 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. Other methods focus on reducing communication complexity or providing additional properties, such as delayed input.

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 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 and the common reference string models. The composition in [ABO+24] uses a dual-mode commitment scheme that preserves the soundness of the underlying protocols but results in computational zero-knowledge protocols. It remains an open problem to devise a multi-round composition method that simultaneously preserves the quality of soundness and zero-knowledge in the plain model as the CDS does for three-move protocols.

Trapdoor commitments and instance dependent commitments. The idea of constructing a commitment scheme from a three-move interactive proofs dates back to [Dam90]. When commitment key x is chosen from no-instances of a language where yes and no instances are indistinguishable, it forms an instance-dependent commitment scheme (IDTC) [CPS+16a]. Our trapdoor commitment can be used as IDTC as well. IDTCs are an essential building block in delayed-input proof systems used in various cryptographic protocols such as [Vis17,CPV20,FHJ20,HV20,ABFV22].

Applications of trapdoor commitments. Trapdoor commitments enjoy various cryptographic applications. Recent ones include constructions of advanced signature schemes such as threshold ring signatures [HS20] and adaptor signatures [LTZ24]. Both rely on a black-box use of trapdoor commitments, which they instantiate based on a generic NP reduction for graph hamiltonicity that involves large practical overheads.

As an example, let us motivate the usefulness of trapdoor commitments for the case of adaptor signatures. An adaptor signature scheme [Poe17] is characterized by a signature scheme and a NP language $\mathcal{L}$. It presents the property that, given a so-called pre-signature $\bar{\sigma}$ and an adapted signature

σ on a message, one can extract an NP witness w for some public instance $x \in \mathcal{L}$. This feature is closely related to the *trapdoor extractability* property of a trapdoor commitment scheme, which says that given two valid opening proofs to different values m, m' for the same commitment *com*, one can extract a valid trapdoor *td*. This is observed in [LTZ24], where messages are pre-signed by signing a commitment *com* and opening *open* to some message m_0. Then, a pre-signature is adapted by trapdoor-opening *com* to some message $m \neq m_0$. Given both openings, the trapdoor of the commitment scheme (which is the witness w) can be retrieved.

By constructing a trapdoor commitment where the trapdoor corresponds to the witness of any NP relation, their framework yields (plausibly post-quantum) adaptor signatures for NP relations. A similar consideration applies to the threshold ring signatures from [HS20]. As we instantiate our trapdoor commitment from an efficient framework such as MPCitH, our result enables more efficient post-quantum secure constructions of these signatures.

2 Preliminaries

2.1 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 $[n]$.

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

For more notation related to the tree of transcript, Let T be a $(k_1, \ldots, k_\mu)$-tree of transcripts. Every node in depth i is indexed by $id := (1, j_1, \ldots, j_{i-1}) \in 1 \times [k_1] \times \cdots \times [k_i]$. 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 $\mathsf{trans}(T, id)$. Let $\mathsf{chal}(T, id)$ denote a set of challenges assigned to outgoing edges from node id. For transcript $\tau := (a, c_1, z_1, \ldots, c_\mu, z_\mu)$, $\mathsf{prefix}(\tau, i)$ denotes the transcript up to i-th node $(a, c_1, z_1, \ldots, c_i, z_i)$.

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 $R(x, w) = 1$.*

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.*

2.3 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 should satisfy the following properties:

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

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

Binding. *A trapdoor commitment scheme is computationally binding if for any PPT adversary $\mathcal{A}$,*

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

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 stateful 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 $m, m' \in \mathcal{M}$ 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.

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*

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

Besides, we say that a trapdoor commitment is offline trapdoor extractable *if there exists an algorithm $\mathsf{Ext}_{\mathsf{off}}$ such that for any $(ck, td) \leftarrow \mathsf{Gen}(1^\lambda)$ and two accepting openings $(com, m, open)$ and $(com, m', open')$ with $m \neq m'$,*

$$\Pr[f_t(ck, td') = 0 \mid td' \leftarrow \mathsf{Ext}_{\mathsf{off}}(ck, com, m, open, m', open')] \leq \mathsf{negl}(\lambda).$$

2.4 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 8 (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 $m \in S$,*

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

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

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

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

3 Critical-Round Zero-Knowledge and Soundness

3.1 Formulation

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

A slightly stronger variation of critical-round zero-knowledge requires a simulator that takes $k-1$ possible challenges for the critical round. Formally:

Definition 10 (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-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.*

In terms of soundness, critical round has different meaning than that of zero-knowledge. Critical-round special soundness captures that the protocol behaves as a round-by-round sound one up to the critical round, and as a proof of knowledge thereafter. The critical round i^*_{zk} in the definition of CRZK is independent of the critical round i^*_{ss} in the special soundness definition. Depending on the specific context, however, a relationship between these two rounds can be established.

Definition 11 (Critical-Round Special Soundness). *A $(2\mu+1)$-move public-coin proof protocol is $(\kappa_1, \ldots, \kappa_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-critical-round special sound (CRSS for short) at round i^*_{ss} if the following properties hold:*

- *For each prefix $\tau_i := \mathsf{prefix}(\tau, i)$, $i \in [i^*_{\mathsf{ss}} - 1]$, of an accepting transcript τ, there exists a set of κ_i bad challenges $\mathcal{BC}_{\tau_i} \in \mathcal{C}_i{}^{\kappa_i}$ where $\kappa_i/|\mathcal{C}_i|$ is negligible in λ.*
- *There exists a polynomial-time algorithm, Ext, that, given any accepting $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree of transcripts whose trunk of transcripts $\tau^* := (a, c_1, z_1, \ldots, c_{i^*_{\mathsf{ss}}-1}, z_{i^*_{\mathsf{ss}}-1})$ satisfies $c_i \notin \mathcal{BC}_{\tau_i}$ for $\tau_i = \mathsf{prefix}(\tau^*, i)$ for all $i \in [i^*_{\mathsf{ss}} - 1]$, outputs w satisfying $R(x, w) = 1$.*

We may also write bad challenges as $\mathcal{BC}_i$ instead of $\mathcal{BC}_{\tau_i}$ when τ_i is clear from the context.

3.2 Relations to Other Notions

In this section we compare related notions to CRSS and CRZK. We also derive concrete bounds for multi-round CRSS interactive protocols, and for non-interactive protocols obtain via the Fiat-Shamir transform.

Generalizations of special soundness. Our CRSS (Definition 11) can be written as prefix-conditioned $\mathfrak{s}$-soundness for a specific prefix-conditioned soundness function $\mathfrak{s}$ relative to an adversary in [HJMN24]. We choose our formulation for its simplicity and intuitiveness, as it best serves our purpose in combination with CRZK.

Any $(k_1, \ldots, k_\mu)$-SS protocol is $(k_1, \ldots, k_\mu)$-CRSS at round 1. Hence CRSS is a seamless generalization of SS. As well as SS, our CRSS can be generalized to handle erroneous transcripts for the $(k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$ part following the formulation of statistical special soundness [ABO+24], predicate special soundness [AAB+24], or $\mathfrak{G}$-soundness [DFMS22] yielding the notion of statistical CRSS, and so on.

CRSS implies knowledge soundness. In each round $i < i^*_{\mathsf{ss}}$, a good challenge can be sampled uniformly from $\mathcal{C}_i$ since $\kappa_i/|\mathcal{C}_i|$ is negligible in λ. In the remaining rounds $i^*_{\mathsf{ss}} \le i \le \mu$, the tree builder behaves essentially the same as a special soundness tree builder in [ACK21]. We now describe our knowledge extractor as follows:

The knowledge extractor $\mathsf{Ext}_{\mathsf{ks}}$, given access to a *deterministic* (dishonest) prover P^* and statement x, first samples the prover's randomness r uniformly at random. It then invokes the tree builder $\mathcal{T}_0(r)$, which is formally described in Figure 1. $\mathcal{T}_0(r)$ first obtains the initial message a from the prover $P^*(x; r)$ and eventually returns a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree of accepting transcripts from recursions. It aborts if the tree builder aborts. The knowledge extractor then invokes the CRSS extractor to extract a witness corresponding to the statement x.

The following lemma modified from [ACK21] gives the expected running time and success probability of the tree builder algorithm $\mathcal{T}$.

Lemma 1. *Let $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu) \to \{0, 1\}$ be the verification predicate corresponding to the $(2\mu + 1)$-move interactive proof system (P, V) and let δ denote the success probability of a (cheating) prover P^* passing the verification on statement x and fixed randomness r. Then, the expected number of invocations to the cheating prover P^* by the $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree builder algorithm $\mathcal{T}$ defined above is at most $\prod_{i=i^*_{\mathsf{ss}}}^{\mu} k_i$. The probability that $\mathcal{T}$ outputs a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree tree is at least*

$$\delta - \sum_{i=i^*_{\mathsf{ss}}}^{\mu} \frac{k_i - 1}{|\mathcal{C}_i|}.$$

Recursive Tree Builder $\mathcal{T}_i(r, c_1, \ldots, c_i) \to (b, \mathsf{tree})$.
Construction.
If $i = \mu$,
- Obtain a transcript $\pi = (a, c_1, z_1, \ldots, c_\mu, z_\mu) \leftarrow P^*(x; r)$ with verifier's challenges $c_1, \ldots, c_\mu$. (Count as 1 query)
- Output $b \leftarrow \mathsf{V}(x, \pi)$.
- If $b = 0$ then abort.
- Else set $\mathsf{tree} = \{\pi\}$ and return (b, tree).

If $i^*_{\mathsf{ss}} - 1 \le i < \mu$,
- Sample $c_{i+1} \leftarrow\!\!\$\ \mathcal{C}_{i+1}$.
- Run $\mathcal{T}_{i+1}(r, c_1, \ldots, c_i, c_{i+1}) \to (b, \mathsf{tree})$.
- Abort if $\mathcal{T}_{i+1}(r, c_1, \ldots, c_i, c_{i+1})$ aborts.
- Else repeat
  - Sample $c'_{i+1} \leftarrow\!\!\$\ \mathcal{C}_{i+1} \setminus \{c_{i+1}\}$ without replacement,
  - Run $\mathcal{T}_{i+1}(r, c_1, \ldots, c_i, c'_{i+1}) \to (b', \mathsf{tree}')$,
  - If $b' = 1$ then append tree' to tree,

  until either $k_{i+1} - 1$ additional tree's have been appended (return $b = 1$ and tree) or until all challenges in $\mathcal{C}_{i+1}$ have been tried (abort).

If $0 \le i < i^*_{\mathsf{ss}} - 1$,
- Sample $c_{i+1} \leftarrow\!\!\$\ \mathcal{C}_{i+1}$.
- Run $\mathcal{T}_{i+1}(r, c_1, \ldots, c_i, c_{i+1}) \to (b, \mathsf{tree})$.
- Abort if $\mathcal{T}_{i+1}(r, c_1, \ldots, c_i, c_{i+1})$ aborts.
- Else return $b = 1$ and tree.

Fig. 1. Recursive Tree Builder $\mathcal{T}_i(r, c_1, \ldots, c_i)$, which is $\mathcal{T}_0(r)$ at $i = 0$.

Proof. (Sketch) Our tree builder behaves essentially the same as the collision game in [ACK21], where the matrix H is replaced with a verification predicate V. In the round $i^*_{\mathsf{ss}} \le i \le \mu$, the above lemma simply follows the analysis of Lemma 5 in [ACK21]. When it comes to the first $i^*_{\mathsf{ss}} - 1$ rounds, the tree building algorithm only samples one element from the challenge space and checks if a series of recursive algorithms successfully return a tree of accepting transcripts, which has no influence on total number of transcripts. Thus the (expected) total number of invocations to P^* is still $\prod_{i=i^*_{\mathsf{ss}}}^{\mu} k_i$. The success probability can also be directly computed from the above analysis. □

Next, we compute the probability that the CRSS extractor succeeds in extracting a witness from a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree output by above tree builder. In each round $1 \le i < i^*_{\mathsf{ss}}$ the tree builder successfully samples a good challenge $c_i \notin \mathcal{BC}_i$ with probability at least $(|\mathcal{C}_i| - \kappa_i)/|\mathcal{C}_i|$. Thus, the success probability for knowledge extractor $\mathsf{Ext}_{\mathsf{ks}}$ is

$$\Pr[w \leftarrow \mathsf{Ext}_{\mathsf{ks}}^{P^*}(x) : R(x, w) = 1] \ge \left(\delta - \sum_{i=i^*_{\mathsf{ss}}}^{\mu} \frac{k_i - 1}{|\mathcal{C}_i|}\right) \left(\prod_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{|\mathcal{C}_i| - \kappa_i}{|\mathcal{C}_i|}\right) \ge \delta - \sum_{i=i^*_{\mathsf{ss}}}^{\mu} \frac{k_i - 1}{|\mathcal{C}_i|} - \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|},$$

where $\delta := \Pr[\langle P^*, V\rangle(x) = 1]$.

Recall the definition of knowledge soundness (Definition 3), $\Pr[w \leftarrow \mathsf{Ext}_{\mathsf{ks}}^{P^*}(x) : R(x, w) = 1] \ge \Pr[\langle P^*, V\rangle(x) = 1] - \epsilon_{\mathsf{ks}}$. It turns out that the knowledge error is given by

$$\epsilon_{\mathsf{ks}} = \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|} + \sum_{i=i^*_{\mathsf{ss}}}^{\mu} \frac{k_i - 1}{|\mathcal{C}_i|}.$$

CRSS implies special soundness. To be precise, $(\kappa_1, \ldots, \kappa_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-CRSS implies $(k_1, \ldots, k_\mu)$-SS for $k_i = \kappa_i + 1$, $i \in [i^*_{\mathsf{ss}} - 1]$ when $\prod_i \kappa_i$ is bound by a polynomial in λ. The extractor for SS can be built by invoking the extractor of CRSS on every $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree in the given $(\kappa_1 + 1, \ldots, \kappa_{i^*_{\mathsf{ss}}-1} + 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree. This strategy always succeeds since in each round $i \in [1, i^*_{\mathsf{ss}} - 1]$ there exists at least one branch where the challenge does not belong to $\mathcal{BC}_{\tau_i}$ as demanded by the CRSS extractor. For the SS extractor to be efficient, we require $\prod_i \kappa_i$ to be bound by a polynomial in λ, which is also the case for a ($(k_1, \ldots, k_\mu)$-special soundness) knowledge extractor efficiently building

a $(k_1, \ldots, k_\mu)$-tree of transcripts. This implication is particularly useful as $(k_1, \ldots, k_\mu)$-SS further implies knowledge soundness and Fiat-Shamir security [ACK21,AFK23]. We note that the condition $\prod_i \kappa_i < poly(\lambda)$ is satisfied by MPCitH-based constructions as we observe in the next section.

CRSS and Fiat-Shamir security. We compute the knowledge error of the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a $(\kappa_1, \ldots, \kappa_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-critical-round special sound interactive proof Π in which the critical round is i^*_{ss}. Let a Q-query cheating prover P^* make Q_i queries at round i such that $\sum_{i=1}^{\mu} Q_i = Q$. We briefly describe the knowledge extractor $\mathsf{Ext}_{\mathsf{fs}}$. In fact, it is essentially the same as in [AFK23], except that the sub-extractor Ext_i at round $i < i^*_{\mathsf{ss}}$ just runs Ext_{i+1} and relays the queries. It aborts if Ext_{i+1} returns 0. From Theorem 2 in [AFK23], the sub-extractor successfully returns an accepting $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree with probability at least

$$\frac{\delta - (\sum_{i=i^*_{\mathsf{ss}}}^{\mu} Q_i + 1)\theta_{i^*_{\mathsf{ss}}}}{1 - \theta_{i^*_{\mathsf{ss}}}}$$

where $\theta_{i^*_{\mathsf{ss}}} = 1 - \prod_{i=i^*_{\mathsf{ss}}}^{\mu}(1 - \frac{k_i - 1}{|\mathcal{C}_i|})$ and δ denotes the success probability of the cheating prover convincing the verifier. The expected number of invocations to P^* is at most $\prod_{i=i^*_{\mathsf{ss}}}^{\mu} k_i + Q \cdot (\prod_{i=i^*_{\mathsf{ss}}}^{\mu} k_i - 1)$.

Next, the knowledge extractor $\mathsf{Ext}_{\mathsf{fs}}$ runs the CRSS extractor with the input of the $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree. This extraction fails if at least one of the responses to queries from the cheating prover in round $i < i^*_{\mathsf{ss}}$ is a bad challenge. We denote this event to be E_a. Suppose that $Q \leq \frac{1}{\sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|}}$, we have

$$\Pr[E_a] \leq \max_{\sum_{i=1}^{i^*_{\mathsf{ss}}-1} Q_i \leq Q} \left(1 - \prod_{i=1}^{i^*_{\mathsf{ss}}-1} \left(1 - \frac{\kappa_i}{|\mathcal{C}_i|}\right)^{Q_i}\right) \leq \max_{\sum_{i=1}^{i^*_{\mathsf{ss}}-1} Q_i \leq Q} \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{Q_i \kappa_i}{|\mathcal{C}_i|} \leq Q \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|}.$$

Finally, we are able to compute the success probability of the knowledge extractor, which is at least

$$\begin{aligned}\Pr[R(x, \mathsf{Ext}_{\mathsf{fs}}^{P^*}(x)) = 1] &\geq \left(\frac{\epsilon - \left(\sum_{i=i^*_{\mathsf{ss}}}^{\mu} Q_q + 1\right)\theta_{i^*_{\mathsf{ss}}}}{1 - \theta_{i^*_{\mathsf{ss}}}}\right)\left(1 - Q \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|}\right) \\ &\geq \frac{\epsilon - (Q+1)\,\theta_{i^*_{\mathsf{ss}}} - Q \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|}}{1 - \theta_{i^*_{\mathsf{ss}}}}.\end{aligned}$$

By the definition of knowledge soundness (Definition 3), the knowledge error

$$\epsilon_{\mathsf{fs}}(Q) = (Q+1)\,\theta_{i^*_{\mathsf{ss}}} + Q \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|} \leq (Q+1)\left(\theta_{i^*_{\mathsf{ss}}} + \sum_{i=1}^{i^*_{\mathsf{ss}}-1} \frac{\kappa_i}{|\mathcal{C}_i|}\right) \leq (Q+1)\,\epsilon_{\mathsf{ks}}$$

where ϵ_{ks} is the knowledge error of the corresponding interactive protocol Π.

CRZK leads to k-zero knowledge. We recap the notion of k-zero-knowledge 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. Formally:

Lemma 2. *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 such that the first message has sufficient min-entropy. Let Π_{FS} be the corresponding non-interactive proof protocol via Fiat-Shamir transform. Then Π_{FS} is k-zero knowledge with $k = i^*_{\mathsf{zk}}$.*

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 $H(x, a, z_1, \ldots, z_{i^*_{\mathsf{zk}}-1}) := c_{i^*_{\mathsf{zk}}}$.
3. Obtain $(st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)$.

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 20) 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.

3.3 Instantiations of Critical-Round Proofs

Critical Rounds in MPCitH-based Proofs. We show that our new notions capture the multi-round proofs based on the MPCitH framework. We cast the honest-verifier zero-knowledge proof in [KKW18] (denoted by KKW) and provide formal proof that their 5-move protocol satisfies CRZK and CRSS.

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. [8]

We show that KKW is CRZK at round $i^*_{\mathsf{zk}} = 2$ and $(1,2)$-CRSS at round $i^*_{\mathsf{ss}} = 2$ as the theorem below. The formal proof is shown in Appendix C together with the detailed description of the KKW framework.

Theorem 1. *The 5-move interactive honest-verifier zero-knowledge argument in [KKW18] (denoted as the KKW protocol), assuming that the hash function used is collision-resistant and the commitment scheme used is computationally binding and hiding, is critical-round special honest-verifier zero-knowledge at round $i^*_{\mathsf{zk}} = 2$ and $(1,2)$-critical-round special sound at round $i^*_{\mathsf{ss}} = 2$.*

Let (a, c_1, z_1, c_2, z_2) be a transcript of the KKW protocol. The intuition behind the proof is described below:

- CRZK at round $i^*_{\mathsf{zk}} = 2$ is proved by constructing two simulators $(\mathsf{SimA}, \mathsf{SimZ})$ that work as $\mathsf{SimA}(x, c_2) \to (st_1, a)$, $\mathsf{SimZ}(st_1, c_1) \to (z_1, st_2)$, and $\mathsf{SimZ}(st_2, c_2) \to z_2$. The first simulator SimA simply generates m preprocessing materials honestly as in the first message of an honest execution. Then, SimZ is built by from the simulator Sim of the MPC protocol. For this, a key observation is that the witness is required only during the MPC execution and not during the preprocessing phase. Hence, the witness is simulatable with Sim given any outputs from an honestly executed preprocessing phase. We note that Sim starts simulating by sampling legitimate preprocessing material and uses it to simulate the online phase, then defines a valid broadcast message in the reconstruction output step. Therefore, we can redefine the input of Sim to consist of honest preprocessing material.
- To see $(1,2)$-CRSS holds at round $i^*_{\mathsf{ss}} = 2$, consider a prefix (a, c_1, z_1). The only way that the prover can cheat is by (1) guessing c_1 correctly and then (2) using the c_1-th preprocessing material to cheat by generating a "fake" view of the parties in MPC protocol in the next move. Therefore, for each $c_1 \in [m]$, there is at most one bad challenge, and hence $|\mathcal{BC}|/|\mathcal{C}|$ is negligible for large enough m and for a sufficiently large number of parallel repetitions. We construct an extractor, Ext, that either extracts an actual witness or finds a collision for the commitment or the hash function. Given two accepted transcripts (a, c_1, z_1, c_2, z_2) and $(a, c_1, z_1, c'_2, z'_2)$ with the same prefix (a, c_1, z_1), the extractor Ext relies on the fact that $c_2 \neq c'_2$. Hence, from (z_2, z'_2), Ext can learn the

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

view of all n parties of the MPC protocol using same preprocessing material that was committed (and not opened) in the first three moves.

Critical Rounds in IOP-based Proofs. We discuss the applicability of our CRZK and CRSS notions to other proof systems of practical interest, in particular SNARKs based on IOPs [RRR16,BCS16] and polynomial IOPs [BFS20]. For these, 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. These protocols can be regarded as $(\kappa_1, \ldots, \kappa_\mu)$-CRSS with critical round $i^*_{\mathsf{ss}} = \mu + 1$. The sets of bad challenges $\mathcal{BC}_i$ at each round are related to the polynomial evaluation checks and batching operations. Usually, these rely on the statistical soundness of the Schwartz-Zippel lemma. Then, the knowledge extractor can be naturally adjusted to the algebraic group model 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. For instance, if one considers a successful adversary for Plonk [GWC19], in the algebraic group model the extractor always obtains either a valid witness or an instance breaking the Q-DLOG assumption over the groups, given that the transcript does not include bad challenges. 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. A recent work by Lipmaa, Parisella, and Siim [LPS24] characterizes Plonk as a computationally special-sound 5-round protocol, revealing the concrete constants κ_i that parametrize its special soundness. The usual variant of Plonk consists of 4 rounds, but the first round is divided into two for their analysis. Precisely, they obtain that for an upper bound of n constraints, Plonk is $\vec{k}$-special sound where $\vec{k} = (3n + 1, 3n + 1, 3, 4n + 21, 6)$.

We notice 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 $(1, 1, 1, k_4, k_5)$-tree of transcripts if the first three challenges are good, i.e., $c_i \notin \mathcal{BC}_i$ for $i = 1, 2, 3$. Hence, we conclude that Plonk satisfies $(\kappa_1 = 3n, \kappa_2 = 3n, \kappa_3 = 2, k_4, k_5)$-CRSS where $i^*_{\mathsf{ss}} = 4$ in the standard model. Unfortunately, k_4 grows linearly with the witness size, which is a drawback for the efficiency of our applications. In our AGM-based analysis, the resulting constants are $\vec{\kappa} = (3n, 3n, 2, 4n + 20, 5)$ for a critical round at $i^*_{\mathsf{ss}} = 6$.

Critical-round zero knowledge also applies to some of these protocols. The most prominent example is again Plonk, which satisfies critical-round zero knowledge at round $i^*_{\mathsf{zk}} = 3$. In short, the main observation is that the first three messages of the Plonk prover consist of commitments to various polynomials, whereas the fourth and the fifth prover messages 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 third 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], and we refer to their work for details.

4 Trapdoor Commitment from Multi-Round Protocol

4.1 Construction and Security

Given a $(2\mu + 1)$-move public-coin proof protocol, $\Pi_{i^*} = (\mathsf{A}, \mathsf{Z}, \mathsf{V}, \mathsf{SimA}, \mathsf{SimZ})$, for relation R that is critical at round i^* (the critical rounds for special HVZK and special soundness are the same $i^*_{\mathsf{ss}} = i^*_{\mathsf{zk}} = i^*$), and two hash functions G and H, we construct a trapdoor commitment scheme $\mathsf{K} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{Equiv}, \mathsf{Ver})$ as follows. For simplicity, we assume that the message space $\mathcal{M} = \mathcal{C}_{i^*}$.

Theorem 2. $\mathsf{K}_{\Pi_{i^*}}$ *in Figure 2 constitutes a trapdoor commitment scheme with the following properties:*

- *It is hiding if Π_{i^*} is critical-round special honest verifier zero-knowledge at round i^*.*
- *It is both binding and trapdoor extractable if Π_{i^*} is perfectly complete, G and H are non-programmable and programmable random oracles, respectively, relation R is one-way, and Π_{i^*} is $(\kappa_1, \ldots, \kappa_{i^*-1}, sk_{i^*}, \ldots, k_\mu)$-critical-round special sound with $k_{i^*} = 2$.*
- *It is equivocal if Π_{i^*} is critical-round honest verifier zero-knowledge at round i^*.*

Set up.
Given a $(2\mu+1)$-move public-coin proof protocol, $\Pi_{i^*} = (\mathsf{A}, \mathsf{Z}, \mathsf{V}, \mathsf{SimA}, \mathsf{SimZ})$, for relation R that is critical at round i^*, and two hash functions G and H.

Construction.

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 $c_{i^*} := m$, $(st_1, a) \leftarrow \mathsf{SimA}(x, c_{i^*})$. For $i = 1, \ldots, \mu$, do

$$c_i := \begin{cases} G(x, a, c_1, z_1, \ldots, c_{i-1}, z_{i-1}) & (i < i^*) \\ c_{i^*} & (i = i^*) \\ H(x, a, c_1, z_1, \ldots, c_{i-1}, z_{i-1}) & (i > i^*) \end{cases} \tag{1}$$

and $(st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i)$.
Output $com := (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$ and $open := (z_{i^*}, c_{i^*+1}, z_{i^*+1}, \ldots, c_\mu, z_\mu)$.

TCom: Given (w, x) as input, compute $(st_1, a) \leftarrow \mathsf{A}(x, w)$, and for $i = 1, \ldots, i^* - 1$, do $c_i := G(x, a, c_1, z_1, \ldots, c_{i-1}, z_{i-1})$ and $(st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i)$.
Output $com := (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$ and st_{i^*}.

Equiv: Given (st_{i^*}, m), compute, for $i = i^*, \ldots, \mu$, $c_i := H(x, a, c_1, z_1, \ldots, c_{i-1}, z_{i-1})$ (except that $c_i := m$ for $i = i^*$), and $(st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i)$.
Output $open := (z_{i^*}, c_{i^*+1}, z_{i^*+1}, \ldots, c_\mu, z_\mu)$.

Ver: Given x, com, m, and $open$ 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 relation in (1), and output $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.

Fig. 2. Trapdoor Commitment Scheme $\mathsf{K}_{\Pi_{i^*}}$

Proof. Correctness is verified by inspection. Hiding property is proved by a game transition argument. We begin with the left term of the hiding definition in Definition 6;

$$\begin{aligned} P_0 &:= \Pr\left[1 \leftarrow \mathcal{A}(ck, com) \,|\, (ck, td) \leftarrow \mathsf{Gen}(1^\lambda), (com, open) \leftarrow \mathsf{Com}(ck, m)\right] \\ &= \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, c_1, z_1, \ldots, & (x, w) \leftarrow L_{RW}(1^\lambda), c_{i^*} = m, \\ \qquad c_{i^*-1}, z_{i^*-1})) & (st_1, a) \leftarrow \mathsf{SimA}(x, c_{i^*}), \\ & \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i) \end{array}\right] \end{aligned}$$

We then modify the game by replacing the zero-knowledge simulator with a real prover algorithm.

$$P_1 := \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, c_1, z_1, \ldots, & (x, w) \leftarrow L_{RW}(1^\lambda), c_{i^*} = m, \\ \qquad c_{i^*-1}, z_{i^*-1})) & (st_1, a) \leftarrow \mathsf{A}(x, w), \\ & \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i) \end{array}\right]$$

Then $|P_0 - P_1|$ is bound by the zero-knowledge error, say ϵ_{crzk}, as in Definition 9. We next replace m to m' and obtain:

$$P_2 := \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, c_1, z_1, \ldots, & (x, w) \leftarrow L_{RW}(1^\lambda), c_{i^*} = m', \\ \qquad c_{i^*-1}, z_{i^*-1})) & (st_1, a) \leftarrow \mathsf{A}(x, w), \\ & \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{Z}(st_i, c_i) \end{array}\right]$$

which obviously hold $P_1 = P_2$ since the view of $\mathcal{A}$ is unchanged. Finally, we make it back to the simulation as follows.

$$P_3 = \Pr\left[\begin{array}{l|l} 1 \leftarrow \mathcal{A}(x, (a, c_1, z_1, \ldots, & (x, w) \leftarrow L_{RW}(1^\lambda), c_{i^*} = m', \\ \qquad c_{i^*-1}, z_{i^*-1})) & (st_1, a) \leftarrow \mathsf{SimA}(x, c_{i^*}), \\ & \forall i \in [\mu],\ (st_{i+1}, z_i) \leftarrow \mathsf{SimZ}(st_i, c_i) \end{array}\right]$$

As before, $|P_2 - P_3|$ is upper bound by ϵ_{crzk}. Since P_3 is the same as the right term of the hiding definition, we obtain $P_0 - P_3 \leq 2\epsilon_{crzk}$ as a bound for the hiding property that is negligible as ϵ_{crzk} is negligible by assumption.

Next, we prove the binding property. Let $\mathcal{A}$ be an algorithm that opens a commitment in two ways with high probability. Precisely, given commitment key x, it outputs, with noticeable probability ϵ, $(com, m_1, open_1, m_2, open_2)$ that satisfies $m_1 \neq m_2$, $m_1, m_2 \in \mathcal{C}_{i^*}$, $\mathsf{Ver}(com, m_j, open_j) = 1$ for $j = \{1, 2\}$. Given $\mathcal{A}$ as a black-box, we construct $\mathcal{B}$ that breaks one-wayness of R. The task of $\mathcal{B}$ is to construct a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-tree of transcripts. $\mathcal{B}$ is given access to non-programmable

random oracle G and simulates programmable random oracle H. Given x, $\mathcal{B}$ runs $\mathcal{A}$ giving x as input. $\mathcal{B}$ behaves transparent for queries to G and does lazy sampling simulation for H. $\mathcal{B}$ obtains a valid colliding opening $(com, m_1, open_1, m_2, open_2)$. Let $com = (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$. Let T be a $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-tree whose nodes and edges are initialized with an empty value. $\mathcal{B}$ assigns $(com, m_1, open_1)$ and $(com, m_2, open_2)$ to two paths on T. It then builds a complete tree of transcripts as follows. It finds a leaf (id_a) of T that no value is assigned yet. It then finds a node, (id_b), on $\mathsf{path}(T, id_a)$ that no values are assigned on the path from (id_b) to (id_a) except for node (id_b). It rewinds $\mathcal{A}$ to the point that it queries $\mathsf{trans}(T, id_b)$ to H and answers with a uniformly chosen distinct challenge c^* that $c^* \notin \mathsf{chal}(T, id_b)$. If $\mathcal{A}$ outputs new valid colliding openings one of which matches $(\mathsf{trans}(T, id_b), c^*, \ldots)$, it assigns the partial transcript $(c^*, \ldots)$ on the path from (id_b) to (id_a). Once all nodes and edges of T are assigned values, $\mathcal{B}$ runs $w \leftarrow \mathsf{Ext}(T)$ and outputs w. This completes the description of $\mathcal{B}$.

We claim that, for $com = (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$, $c_i \neq \mathcal{BC}_{id}$ holds for id of node at each depth $i \in [i^* - 1]$. To see this, observe that c_i is determined by query $G(a, c_1, z_1, \ldots, c_{i-1}, z_{i-1})$. The input to G first determines $\mathcal{BC}_{id}$ and then c_i is randomly picked. Since Π_{i^*} is critical-round special sound at i^*, the probability that c_i is in $\mathcal{BC}_{id}$ is negligibly small for every $i \in [i^* - 1]$.

We next claim that, in the rewinding search, $\mathcal{A}$ outputs valid colliding openings one of which contains $(\mathsf{trans}(T, id_b), c_{j+1})$ with sufficiently large probability. This is due to the standard rewinding argument in the programmable random oracle model. This completes the proof of binding, and also the proof of trapdoor extractability as $\mathcal{B}$ actually extracts a valid witness w such that $R(x, w) = 1$.

Finally, equivocability holds due to the critical-round honest verifier zero-knowledge of Π_{i^*} since the equivocability game in Definition 6 is exactly the critical-round zero-knowledge game in Definition 9 where TCom runs A and Z honestly up to $i^* - 1$ and Equiv runs $\mathsf{Z}(st_i, c_i)$ for i^* and further rounds for arbitrary challenge c_{i^*} where the message is embedded. □

Removing random oracles. For a subclass of Π_{i^*}, we can remove random oracles from the above construction. First, for Π_{i^*} with $k_i = 1$ for all $i < i^*$, hash function G can be replaced with a correlation intractable hash function, which can be constructed from falsifiable assumptions [CLW18,PS19]. Removing H is more difficult as we essentially rely on its programmability for building a complete subtree. Since the binding game is non-interactive, without a random oracle, there is no way to obtain more transcripts to fill unassigned branches of the tree. Nevertheless, the obvious case is that $i^* = \mu$, i.e., the last round is critical and there are no further challenges. Several multi-round protocols, e.g. [KKW18,BN20,KZ22,FJR22], that follow the MPCitH with preprocessing paradigm indeed fall into this class. We also conjecture that, for all 5-move interactive proofs based on the MPCitH paradigm that follow the KKW framework, we have $i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}} = 2$, i.e., the second challenge round is CRZK and CRSS provided that the first challenge round has an appropriate choice of the set $\mathcal{BC}$.

Interactive Commitment Scheme. It is also possible to completely eliminate G and H by making the commitment and opening interactive. In the committing phase, the committer and the receiver engage in the proof protocol up to $2(i^* - 1) + 1$ step where the committer sends z_{i^*-1} to the receiver. The opening phase starts by sending m from the committer and the protocol continues to the final step. This interactive commitment scheme is however only privately verifiable to the receiver engaging in both commitment and opening phases.

Differing critical rounds. For simplicity, our construction focuses on the case $i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}} = i^*$, but the construction and the proof also work in a more general case $i^*_{\mathsf{zk}} \geq i^*_{\mathsf{ss}}$, where we base on $(\kappa_1, \ldots, \kappa_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_{i^*_{\mathsf{zk}}}, \ldots, k_\mu)$-CRSS. A commitment to m is a transcript from round 1 to round $i^*_{\mathsf{zk}} - 1$, $com = (a, c_1, z_1, \ldots, c_{i^*_{\mathsf{zk}}-1}, z_{i^*_{\mathsf{zk}}-1})$, where a is generated by $\mathsf{SimA}(x, m)$. In the proof of binding, rewinding the adversary starts from round i^*_{ss} to obtain a complete tree of transcripts. Note that a programmable random oracle is required at all rounds $i > i^*_{\mathsf{ss}}$. The proof of hiding remains intact.

Interestingly, in the other general case $i^*_{\mathsf{zk}} < i^*_{\mathsf{ss}}$ where we base on $(\kappa_1, \ldots, \kappa_{i^*_{\mathsf{zk}}}, \ldots, \kappa_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-CRSS, the above variant is no longer binding since the adversary would take messages m and m' from the set of bad challenges so that the extractor fails. If $\kappa_{i^*_{\mathsf{zk}}} = 1$, however, it remains binding since either m or m' must be a good challenge that forms a $(1, \ldots, 1, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$-tree of transcripts that suffices for the extractor.

Offline trapdoor extraction. Our construction satisfies trapdoor extractability, but for applications such as adaptor signatures, offline trapdoor extractability is useful. If the critical round $i^*_{\mathsf{ss}} = \mu$ (such

as in the MPCitH-based instantiation, see Section 3.3), then our trapdoor commitment is offline trapdoor extractable, as from a binding break the challenger directly obtains a complete tree of transcripts, and so can extract the trapdoor without having to rewind the adversary.

For arbitrary i^*, this is also possible with a variant of our scheme. The idea is that openings should contain an entire sub-tree of transcripts starting at level $i^* + 1$. If the protocol Π_{i^*} is $(\kappa_1, \ldots, \kappa_{i^*-1}, 2, k_{i^*+1}, \ldots, k_\mu)$ special-sound, then the Com algorithm works as follows:

- For $i = 1, \ldots, i^*$, do as in Figure 2. The commitment is $com = (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$ and the committed message m is used for simulation as $c_{i^*} = m$.
- For all remaining $i = i^* + 1, \ldots, \mu$, set $c_i^{(j_{i^*+1}, \ldots, j_i)}$ for all $j_i = 1, \ldots, k_i$ to be arbitrary pairwise distinct challenges. Then, set $(st_i^{(j_{i^*+1}, \ldots, j_i)}, z_i^{(j_{i^*+1}, \ldots, j_i)}) \leftarrow \mathsf{SimZ}(st_{i-1}^{(j_{i^*+1}, \ldots, j_i)}, c_i^{(j_{i^*+1}, \ldots, j_i)})$ for each j_i.
- An opening *open* contains all challenges $c_i^{(j_{i^*+1}, \ldots, j_i)}$ and responses z_{i^*} and $z_i^{(j_{i^*+1}, \ldots, j_i)}$ for every $i = i^* + 1, \ldots, \mu$ and for every $j_i = 1, \ldots, k_i$.

Clearly, a full tree of transcripts is obtained given two valid openings of $com = (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$ to different messages. Note that with this modification the construction does not need a random oracle after round i^*. The drawback of this approach is the additional opening size and prover time, hence we do not present it as our main construction.

4.2 Extension: Accumulator

We generalize our construction of a trapdoor commitment scheme (Figure 2) 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 3.

Recall that critical-round k_{i^*}-special HVZK (Definition 10) requires that there exists a zero-knowledge simulator $(st_1, a) \leftarrow \mathsf{SimZX}(x, c_{i^*}^{(1)}, \ldots, c_{i^*}^{(k-1)})$ where each $c_{i^*}^{(j)} \in \mathcal{C}_{i^*}$, i.e., that takes $k_{i^*} - 1$ challenges to simulate the first message of the protocol a.[9] 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 3. *The construction $\mathsf{ACC}_{\Pi_{i^*}}$ in Figure 3 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^*. It is binding if G and H are non-programmable and programmable random oracles, respectively, relation R is one-way, and Π_{i^*} is $(\kappa_1, \ldots, \kappa_{i^*-1}, k_{i^*}, \ldots, k_\mu)$-critical-round special sound with $k_{i^*} \geq 3$.*

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 $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$ tree of transcripts. The proof follows a similar strategy as the proof of Theorem 2, 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

[9] 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 critical-round k_{i^*}-special HVZK and CRSS at round i^* for some $k_{i^*} \geq 3$.
- 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 $ck := x$, and the trapdoor is $td := w$.

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

$$c_i := G(x, a, c_1, z_1, \ldots, c_{i-1}, z_{i-1}) \tag{2}$$

  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

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

  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 (2) and (3), and output $\mathsf{V}(x, a, c_1, z_1, \ldots, c_\mu, z_\mu)$.

Fig. 3. Accumulator $\mathsf{ACC}_{\Pi_{i^*}}$

$\pi^* = (m^*, z_j^*, c_{j+1}^*, \ldots, c_\mu^*, z_\mu^*)$. 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 $c_{i^*}^{(j)} = m_j$ for every $j = 1, \ldots, k_{i^*} - 1$ (recall that the accumulator 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 full 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 2. Once T is obtained, $\mathcal{B}$ runs $w \leftarrow \mathsf{Ext}(T)$ and returns w.

It remains to show that the tree of transcripts satisfies the conditions of Definition 11. This again follows as in the previous proof, as for $com = (a, c_1, z_1, \ldots, c_{i^*-1}, z_{i^*-1})$, $c_i \neq \mathcal{BC}_{id}$ holds for every node id of T at any depth $i \in [i^* - 1]$ except with negligible probability. □

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 $k_{i^*} = 2$ from the previous section. 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 3.
- 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 3.

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 3. 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 $j = 1, \ldots, t$.

4.3 Applications to Advanced Signatures

Previous works [HS20] and [LTZ24] that introduce generic constructions of adaptor signatures (AS) for arbitrary NP relations and threshold ring signatures (TRS), respectively, explicitly pose the open question of how to implement post-quantum trapdoor commitments without the overhead from generic NP reductions. In particular, they ask whether these could be built from MPCitH. We resolve both open questions in the affirmative by instantiating their generic compilers with our (standard) trapdoor commitment from Theorem 2, which satisfies all the required properties, and which we can instantiate from MPCitH following Theorem 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[10] for any NP relation via the three-move interactive protocol for graph hamiltonicity, at the cost of a large practical overhead. We remark that our MPCitH-based trapdoor commitment scheme satisfies offline trapdoor extractability as discussed in Section 4.1.

Threshold ring signatures. (t, N)-threshold ring signatures [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 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.

5 Composition of Multi-Round Protocols

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. These relations can be logically composed using a monotone access structure. Thus, the prover will know valid witnesses for some of the relations

[10] 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.

(generating qualified transcripts) and will need to simulate the remaining non-qualified transcripts. In brief, given multi-round proofs that are CRSS and CRZK at critical round i^* ($i^*_{\mathsf{zk}} = i^*_{\mathsf{ss}} = i^*$), 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 in step $2i^*$, the prover uses a secret sharing scheme to generate remaining challenges for the qualified transcripts.

5.1 Construction and Security

Our composition is presented in Figure 4. The private input to the prover algorithm is a set of witnesses $\{w^j\}_{j\in A}$ that $A \in \Gamma$. We assume the reader is familiar with CDS composition and its related definitions, otherwise, a formalization of the CDS composition for multi-round proofs is provided in Appendix A. The ingredients of our protocol include a smooth secret sharing scheme SSS_{Γ^*} and n critical-round proof protocols $\{\Pi^j\}_{j\in[n]}$ with critical round i^*.

Composition Compiler $\Pi^{\mathsf{comp}}_{\Gamma}$.

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

Witness: $\{w^j\}_{j\in A\in\Gamma}$.

Step 1 : Prover computes as follows:

- **for all** $j \in \bar{A}$: Sample $c^j_{i^*} \leftarrow\!\!\$\ \mathcal{C}_{i^*}$; Simulate $(st^j_1, a^j) \leftarrow \mathsf{SimA}^j(x^j, c^j_{i^*})$.
- **for all** $j \in A$: Compute $(st^j_1, a^j) \leftarrow \mathsf{A}^j(x^j, w^j; r^j)$.
- Send $\{a^j\}_{j\in[n]}$ to the verifier.

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

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

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

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

- Compute $\{c^j_{i^*}\}_{j\in[n]} \leftarrow \mathsf{Complete}_{\Gamma^*}(c_{i^*}, \{c^j_{i^*}\}_{j\in\bar{A}})$.
- **for all** $j \in \bar{A}$: Compute $(st^j_{i^*+1}, z^j_{i^*}) \leftarrow \mathsf{SimZ}^j(st^j_{i^*}, c^j_{i^*})$.
- **for all** $j \in A$: Compute $(st^j_{i^*+1}, z^j_{i^*}) \leftarrow \mathsf{Z}^j(st^j_{i^*}, c^j_{i^*})$.
- Send $\{c^j_{i^*}, z^j_{i^*}\}_{j\in[n]}$ to the verifier.

Verification : Accept if and only if

- **for all** $j \in [n]$: $\mathsf{V}^j(x^j, \pi^j = (a^j_1, c^j_1, z^j_1, \ldots, c^j_\mu, z^j_\mu)) = 1$.
- $\mathsf{Checkshares}_{\Gamma^*}(c_{i^*}, \{c^j_{i^*}\}_{j\in[n]}) = 1$.

Fig. 4. Our composition $\Pi^{\mathsf{comp}}_{\Gamma}$ for multi-round protocols with critical round i^*.

Theorem 4 ($\Pi^{\mathsf{comp}}_{\Gamma}$). *If SSS_{Γ^*} is a smooth secret sharing scheme for access structure Γ^*, and every Π^j for relation R^j is complete, $(\kappa_1, \ldots, \kappa_{i^*-1}, k_{i^*}, \ldots, k_\mu)$-special sound and special honest verifier zero-knowledge with critical round i^*, where $\kappa_i = |\mathcal{BC}_i|$ and $k_{i^*} = 2$, then, protocol $\Pi^{\mathsf{comp}}_{\Gamma}$ is a $(2\mu+1)$-move proof system for relation $R_\Gamma((x^1, \ldots, x^n), (w^1, \ldots, w^n)) := \{\exists A \in \Gamma, \forall j \in A,\ R^j(x^j, w^j) = 1\}$. It is complete, $(\kappa'_1, \ldots, \kappa'_{i^*-1}, k_{i^*}, \ldots, k_\mu)$-special sound, and special honest verifier zero-knowledge with critical round i^*, where $\kappa'_i = |\bigcup_{j\in[n]} \mathcal{BC}^j_i|$ and $k_{i^*} = 2$.*

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

Critical-round special soundness. We prove that:

1. In round $i \in [i^* - 1]$, $\mathcal{BC}_i = \bigcup_{j\in[n]} \mathcal{BC}_i^j$;
2. Given any accepting $(1, \ldots, 1, k_{i^*}, \ldots, k_\mu)$-tree of transcripts as input where $c_i \notin \mathcal{BC}_i$ for $i < i^*$, there exists an efficient extractor Ext that outputs a valid set of witnesses $\mathbf{w} = \{w^j\}_{j\in A}$ where $A \in \Gamma$.

Statement 1 is simply proven by the union bound. We have $\kappa'_i = |\mathcal{BC}_i| < n\kappa_i$. Next we move on to construct the extractor for the rounds after the critical round. The composed accepting tree of transcripts can be separated into n split transcripts, which should have the following properties:

- In the critical round i^*, the consistency of the shared challenges can be checked by the $\mathsf{CheckShares}$ algorithm of the secret sharing scheme.
- In non-critical rounds $i^* < i \leq \mu$, the challenge sets of all split transcripts for $j \in [n]$ remain the same as the main challenge.

The extractor Ext just runs extractors Ext_{Π_j} to extract witnesses for all $j \in A$. In non-critical rounds $i^* < i \leq \mu$, since the main challenges are distinct for all k_i branches, the challenges in split transcripts are also distinct among all branches.

In the critical round i^*, the proof is similar to the proof in original CDS composition. The perfect secret sharing scheme on Γ^* and $k_{i^*} = 2$ guarantees that the set A of witnesses $\mathbf{w} = \{w^j\}_{j\in A}$ we extract is a qualified set in Γ. We use the following proposition.

Proposition 1 ([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^*.*

Suppose that the cheating prover successfully responds to two different challenges c and c' in round i^*. We denote the shared challenges for n statements to be $\{c^j\}_{j\in[n]}$ and $\{c'^j\}_{j\in[n]}$. Since $c \neq c'$, we know that for every qualified set $B \in \Gamma^*$, there exists at least one index $j \in B$ such that $c^j \neq c'^j$ from which a witness w_j can be extracted as $k_{i^*} = 2$ and the remaining non-critical round split challenges are distinct among all branches. Since the set A of witnesses we extract has at least one index in every qualified set $B \in \Gamma^*$, we can conclude from Proposition 1 that A is a qualified set in Γ.

Critical-round special Honest verifier zero-knowledge. A special honest verifier zero-knowledge simulator can be constructed straightforwardly based on the facts that Π^j for all $j \in [n]$ are critical-round special honest verifier zero-knowledge at the same round and the smoothness of the secret sharing scheme. □

5.2 Extensions

Case of $i^*_{\mathsf{zk}} \neq i^*_{\mathsf{ss}}$. In the composition in Figure 4, critical-round i^* refers to the zero-knowledge critical round i^*_{zk} relative to the zero-knowledge simulator. Critical-round i^*_{ss} for special soundness can be arbitrary chosen from 1 to $\mu + 1$ without affecting the construction, as long as $k_{i^*_{\mathsf{zk}}} = 2$ when $i^*_{\mathsf{zk}} \geq i^*_{\mathsf{ss}}$ or $\kappa_{i^*_{\mathsf{zk}}} = 1$ when $i^*_{\mathsf{zk}} < i^*_{\mathsf{ss}}$. Having i^*_{ss} in a later round, e.g. in the final one, will benefit in reducing the cost of building the tree of transcripts in the security proof. However, the composition does not preserve the parameters; the composed protocol will have special soundness $(\kappa'_1, \ldots, \kappa'_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu) = (n\kappa_1, \ldots, n\kappa_{i^*_{\mathsf{ss}}-1}, k_{i^*_{\mathsf{ss}}}, \ldots, k_\mu)$. On the other hand, if i^*_{ss} is an early round, e.g. the first one, the composition preserves the parameters as $(k'_1, \ldots, k'_\mu) = (k_1, \ldots, k_\mu)$ in exchange for the larger cost of building the tree of transcripts.

Case of $k_{i^*} > 2$ (or $\kappa_{i^*_{\mathsf{zk}}} > 1$). The above composition only works when the critical round i^* is only 2-special sound, where a secret sharing scheme is applied. This restriction is inherent in the CDS paradigm, since the cheating prover may be able to simulate up to $k_{i^*} - 1$ challenges for each atomic statement beforehand and satisfy the constraint with the verifier's challenges by cleverly combining preselected challenges. (In the worst case, if $(k_{i^*} - 1)^n > |\mathcal{C}_{i^*}|$ for n atomic statements, the cheating prover always succeeds.) Fortunately, such restriction can be avoided via the Share-then-Hash method proposed by Abe et al. [AAB+20] with the help of a random oracle $H : \{0,1\}^* \mapsto \mathcal{C}_{i^*}$. Intuitively, rather than straightly use sampled values $s^j_{i^*}$ as challenges $c^j_{i^*} = s^j_{i^*}$ for simulated transcripts, the prover is now required to obtain challenges $c^j_{i^*} \leftarrow H(x^j, s^j_{i^*})$ from the random oracle.

5.3 Further Applications

Trapdoor commitments with flexible trapdoor allocation. Observe that the composition in Section 5.1 is closed, 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 4.

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 base on a critical-round proof for NP to achieve the same result. This modular approach is particularly useful when the relation in question is not inherently closed under logical composition.

6 Conclusion

We have introduced a novel characterization for multi-round proof protocols, which we name critical-round zero-knowledge and critical-round special soundness. In our analysis, we demonstrate that this class of multi-round protocols is as useful as three-move protocols in applications where a zero-knowledge simulator is used as part of the construction. We also show that our characterization applies to protocols of practical relevance.

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. This raises the question: could this observation be extended into a more general theorem?
- Our security analysis focuses on the classical random oracle model. Although our CRSS implies special soundness for certain parameters, which eventually implies Fiat-Shamir soundness in the quantum random oracle model (QROM) [BGTZ23], we still expect that a careful QROM analysis yields better security bounds than existing analyses for generic multi-round protocols, since no programmability of the random oracle is required before the critical round in our applications.
- It would be of interest to derive concrete security bounds for our trapdoor commitment scheme. We note that by instantiating the scheme from MPCitH, the construction could lead to practical post-quantum secure adaptor and threshold ring signatures. The performance of these schemes largely depends on parameter selection, which also remains as future work.

Acknowledgements

Dung Bui is supported by DIM Math Innovation 2021 (N°IRIS: 21003816) from the Paris Mathematical Sciences Foundation (FSMP) funded by the Paris Ile-de-France Region.
David Balbás is supported by the PICOCRYPT project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101001283), partially supported by projects PRODIGY (TED2021-132464B-I00) and ESPADA (PID2022-142290OB-I00) funded by MCIN/AEI/10.13039/501100011033/ and the European Union NextGenerationEU / PRTR, 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.

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.

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 *CRYPTO 2024*, 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.

ACFY24. G. Arnon, A. Chiesa, G. Fenzi, and E. Yogev. STIR: Reed-solomon proximity testing with fewer queries. In *Advances in Cryptology – CRYPTO 2024, Part X*, *Lecture Notes in Computer Science* 14929, pages 380–413, Santa Barbara, CA, USA, August 18–22, 2024. 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.

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.

AGH+23. C. Aguilar-Melchor, N. Gama, J. Howe, A. Hülsing, D. Joseph, and D. Yue. The return of the SDitH. In *Advances in Cryptology – EUROCRYPT 2023, Part V*, *Lecture Notes in Computer Science* 14008, pages 564–596, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

AHIV17. S. Ames, C. Hazay, Y. Ishai, and M. Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 2087–2104, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press.

BBHR18. E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *ICALP 2018: 45th International Colloquium on Automata, Languages and Programming*, *LIPIcs* 107, pages 14:1–14:17, Prague, Czech Republic, July 9–13, 2018. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

BCC+24. D. Bui, E. Carozza, G. Couteau, D. Goudarzi, and A. Joux. Short signatures from regular syndrome decoding, revisited. Cryptology ePrint Archive, Report 2024/252, 2024.

BCR+19. E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward. Aurora: Transparent succinct arguments for R1CS. In *Advances in Cryptology – EUROCRYPT 2019, Part I*, *Lecture Notes in Computer Science* 11476, pages 103–128, Darmstadt, Germany, May 19–23, 2019. Springer, Cham, Switzerland.

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.

BDK+21. C. Baum, C. Delpech de Saint Guilhem, D. Kales, E. Orsini, P. Scholl, and G. Zaverucha. Banquet: Short and fast signatures from AES. In *PKC 2021: 24th International Conference on*

Theory and Practice of Public Key Cryptography, Part I, *Lecture Notes in Computer Science* 12710, pages 266–297, Virtual Event, May 10–13, 2021. 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.

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.

CCJ23. E. Carozza, G. Couteau, and A. Joux. Short signatures from regular syndrome decoding in the head. In *Advances in Cryptology – EUROCRYPT 2023, Part V*, *Lecture Notes in Computer Science* 14008, pages 532–563, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

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.

CFF+21. M. Campanelli, A. Faonio, D. Fiore, A. Querol, and H. Rodríguez. Lunar: A toolbox for more efficient universal and updatable zkSNARKs and commit-and-prove extensions. In *Advances in Cryptology – ASIACRYPT 2021, Part III*, *Lecture Notes in Computer Science* 13092, pages 3–33, Singapore, December 6–10, 2021. Springer, Cham, Switzerland.

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.

CLW18. R. Canetti, A. Lombardi, and D. Wichs. Fiat-Shamir: From Practice to Theory, Part II (NIZK and Correlation Intractability from Circular-Secure FHE), 2018.

COS20. A. Chiesa, D. Ojha, and N. Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In *Advances in Cryptology – EUROCRYPT 2020, Part I*, *Lecture Notes in Computer Science* 12105, pages 769–793, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland.

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.

CPV20. M. Ciampi, R. Parisella, and D. Venturi. On adaptive security of delayed-input sigma protocols and fiat-shamir NIZKs. In *SCN 20: 12th International Conference on Security in Communication Networks*, *Lecture Notes in Computer Science* 12238, pages 670–690, Amalfi, Italy, September 14–16, 2020. Springer, Cham, Switzerland.

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.

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.

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.

HJMN24. A. Hülsing, D. Joseph, C. Majenz, and A. K. Narayanan. On round elimination for special-sound multi-round identification and the generality of the hypercube for MPCitH. In *Advances in Cryptology – CRYPTO 2024, Part I*, *Lecture Notes in Computer Science* 14920, pages 373–408, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

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.

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

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

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. To appear in Asiacrypt 2024, 2024.

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

PS19. C. Peikert and S. Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In *Advances in Cryptology – CRYPTO 2019, Part I*, *Lecture Notes in Computer Science* 11692, pages 89–114, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

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.

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):1757–1768, sep 2006.

Vis17. I. Visconti. Delayed-input cryptographic protocols. In *13th Conference on Computability in Europe, CiE 2017*, *Lecture Notes in Computer Science* 10307, pages 112–115. Springer, 2017.

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.

Supplementary Material

A CDS Composition

A.1 Monotone Access Structure

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

Definition 12 (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 13 (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 $(\Gamma^*)^* = \Gamma$.

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 $\{s^j\}_{j\in M}$.

$\mathsf{Rec}_\Gamma(\{s^j\}_{j\in A}) \to s$: 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.

$\mathsf{Complete}_\Gamma(s, \{s^j\}_{j\in \bar{A}})$: takes shares of a non-qualified set of shares $\{s^j\}_{j\in \bar{A}}$ for $A \in \Gamma$ and secret s, and outputs $\{s^j\}_{j\in A}$ 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 $A \in \Gamma$, $s \in S$, and $\{s^j\}_{j\in M} \leftarrow \mathsf{Share}_\Gamma(s)$, the distribution of $\{s^j\}_{j\in \bar{A}}$, denoted by $S_{\bar{A}}$, 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 $A \in \Gamma$, $s \in S$, $\{s^j\}_{j\in \bar{A}} \leftarrow S_{\bar{A}}$, and $\{s^j\}_{j\in A} \leftarrow \mathsf{Complete}_\Gamma(s, \{s^j\}_{j\in \bar{A}})$, it holds that $1 \leftarrow \mathsf{CheckShares}_\Gamma(s, \{s^j\}_{j\in M})$.

Efficient SSS_Γ exists for Γ being a threshold structure [Sha79], monotone circuit [BL90], and monotone span program [CDM00]. If, for every $A \in \Gamma$, $S_{\bar{A}}$ equals uniform distribution over $S^{|\bar{A}|}$, then it is called a smooth perfect secret sharing scheme. Shamir's secret sharing scheme for threshold structures is an example.

A.3 CDS Composition

We describe CDS composition [CDS94] in a general form for $2\mu + 1$-move protocols. It matches the original one at $\mu = 1$. We warn that it is for introducing consistent notations for succeeding sections, and indeed not sound for $\mu \geq 2$ and $k(> 3)$-special sound as mentioned earlier.

Let Π_{R^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 honest verifier zero-knowledge simulator of Π_{R^j} by Sim^j. Given Π_{R^j} for $j \in [n]$ and access structure Γ, 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 $j \in \bar{A}$, Prover calls the honest verifier zero-knowledge simulator $(a^j, \{c_i^j\}_{i\in[\mu]}, \{z_i^j\}_{i\in[\mu]}) \leftarrow \mathsf{Sim}^j(x^j)$. 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 $c_i \xleftarrow{\$} \mathcal{C}_i$, and sends it to Prover.

Step $2i+1$. Prover completes shares by $\{c_i^j\}_{j\in[n]} \leftarrow \mathsf{Complete}_{\Gamma^*}(c_i, \{c_i^j\}_{j\in\bar{A}})$, 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 $\{\{c_i^j\}_{j\in[n]}, \{z_i^j\}_{j\in[n]}\}$ 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 $\mathsf{CheckShares}_{\Gamma^*}(c_i, \{c_i^j\}_{j\in[n]})$ for $i \in [\mu]$, and outputs 1 if all outputs are 1, outputs 0, otherwise.

It is shown in [CDS94] that, if every Π_{R^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. It admits offline simulation since zero-knowledge simulator $\mathcal{S}^j$ is invoked only in the first step of the prover algorithm. If every Π_{R^j} is special honest verifier zero-knowledge, the above can be augmented to accept access structure Γ that admits a semi-smooth secret sharing scheme. It is done by modifying the prover's first step in a way that it first shares a random secret to obtain challenge c^j for $j \in \bar{A}$ and runs the special honest verifier zero-knowledge simulator on input x^j and c^j.

B Other Definitions

Definition 14 (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 2.2, 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

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

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 15 (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 statistical soundness errror ϵ_{stss} if there exists a polynomial-time algorithm and a function, ϵ_{stss}, defined over the challenge space, $\{\mathcal{C}_i\}_{i\in[\mu]}$, that, given a distinct $(k_1, \ldots, k_\mu)$-tree of accepting transcripts, outputs w that satisfies $R(x, w) = 1$ with probability at least $1 - \epsilon_{\mathsf{stss}}$. The probability is taken over the choices of challenges. 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 16 (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 17 (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 18 (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 19 ($\mathfrak{G}$-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 20 (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.*

C KKW Framework

C.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 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. Note that these shares are independent of w.

– The masked shares $\{\hat{z}_\alpha\}_{\alpha\in C}$ and broadcast messages (denoted by msgs_i) of each of the parties allow for a *deterministic* simulation 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. 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 $\{\mathsf{state}_{j,i}\}_{i\in[n]}$ together with randomness values (we omit these values for simplicity).
   Later, prover commits to each state as $com_{j,i} = \mathsf{Com}(\mathsf{state}_{j,i})$, each seed_j as $h_j = H(\mathsf{state}_{j,1}, \ldots, \mathsf{state}_{j,c})$.
   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 gets the initial masked values $\{\hat{z}_\alpha\}$ 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.

C.2 Critical Round in KKW

Theorem 5. *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 critical-round special honest-verifier zero-knowledge at round $i^*_{\mathsf{zk}} = 2$ and $(1,2)$-critical-round special sound at round $i^*_{\mathsf{ss}} = 2$.*

Proof. We denote a transcript of this protocol by (a, c_1, z_1, c_2, z_2). The critical-round special HVZK property is proven by constructing two simulators $(\mathsf{SimA}, \mathsf{SimZ})$. 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 $\mathsf{Sim}(\{\mathsf{state}_i\})_{i\neq c_2}, \{\hat{z}_\alpha\}) \rightarrow \mathsf{msgs}_{c_2}$. Now, we define the simulators $(\mathsf{SimA}, \mathsf{SimZ})$ as follows, for challenges $c_1 \leftarrow\!\$\ [m], c_2 \leftarrow\!\$\ [n]$.

– $\mathsf{SimA}(x, c_2) \rightarrow (st_1, a)$: First, execute the first round honestly, i.e., randomly sample m executions of the preprocessing phase $\{\mathsf{seed}_j\}_{j\in[m]}$, then compute $\{\mathsf{seed}_{j,i}\}_{i\in[n]}$, $\{com_{j,i}\}_{i\in[n]}, \{h_j\}$ and h. Output $st_1 := (\{\mathsf{seed}_j\}_{j\in[m]})$ and $a := h$.
– $\mathsf{SimZ}(st_1, c_1) \rightarrow (z_1, st_2)$: To simulate the third message, SimZ simply retrieves the seeds from st_1 and outputs $z_i := \{\mathsf{seed}_j\}_{j\neq c_1}$ and $st_2 := \mathsf{seed}_{c_1}$.

– $\mathsf{SimZ}(st_2, c_2) \to z_2$: To simulate the fifth message, given $st_2 = \mathsf{seed}_{c_1}$, first compute $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$ along with com_{c_1,c_2}, then sample $\{\hat{z}_\alpha\}$, and finally run the MPC simulator $\mathsf{Sim}(\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, \{\hat{z}_\alpha\}) \to \mathsf{msgs}_{c_2}$. Finally, output the value $z_2 := (\{\mathsf{state}_{c_1,i}\}_{i\neq c_2}, \{\hat{z}_\alpha\}, com_{c_1,c_2}, \mathsf{msgs}_{c_2})$.

Due to the security of the MPC protocol against all-but-one corruptions, 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.

We now argue about critical-round $(1, 2)$-special soundness at round $i^*_{\mathsf{ss}} = 2$. Given a $(1, 2)$-tree of the transcript T, all transcripts in T share the same prefix (a, c_1, z_1). Our goal is to show that if there are two accepting transcripts (a, c_1, z_1, c_2, z_2) and $(a, c_1, z_1, c'_2, z'_2)$ with the same prefix (a, c_1, z_1) and $c_1 \notin \mathcal{BC}$ then there exists an extractor Ext that can either extract an actual witness, or find a collision on the commitment scheme or the hash function.

To see this, note that the prefix (a, c_1, z_1) where $c \in [m]$ commits to all m executions of the preprocessing phase and then opens all of them except for the c_1-th. Hence, the only way prover can cheat is by (1) guessing c_1 correctly and then (2) using the c_1-th preprocessing material to cheat by generating "fake" view of parties in MPC protocol. Otherwise, the transcripts sent by the prover cannot pass the verification checks from the verifier in the final round. Therefore, for each $c_1 \in [m]$ there is only one bad challenge defined by the prover from guessing c_1, and hence $|\mathcal{BC}|/|\mathcal{C}|$ is negligible for large enough m and for a sufficiently large number of parallel repetitions. The extractor Ext works as follows:

– From z_1, Ext learns $\{\mathsf{seed}\}_{j\neq c_1}$ and $\{\hat{z}_\alpha\}$.
– Since $c_2 \neq c'_2$ then from (z_2, z'_2), Ext learns all $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$ along with $(com_{c_1,c_2}, \mathsf{msgs}_{c_2}, com_{c_1,c'_2}, \mathsf{msgs}_{c'_2})$.
– From $\{\mathsf{state}_{c_1,i}\}_{i\in[n]}$, Ext learns the value λ_α for each wire α, therefore from $\{\hat{z}_\alpha\}$, Ext can effectively compute a witness w.

We highlight that as $\{\hat{z}_\alpha\}_{i\in[n]}$ 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 unless there is a break in the binding of the commitment scheme or the collision resistance of H (and in that case, the extractor outputs the collision). Otherwise, we have that $\{\mathsf{state}_{c_1,i}, \mathsf{msgs}_i\}_{i\in[n]}$ are generated honestly since $c_1 \notin \mathcal{BC}$. Therefore, $C(w) = 1$ and w is a valid witness. □