On the Security of Nova Recursive Proof System

Hyeonbum Lee and Jae Hong Seo*

Department of Mathematics & Research Institute for Natural Sciences,
Hanyang University, Seoul 04763, Republic of Korea
{leehb3706, jaehongseo}@hanyang.ac.kr

Abstract. Nova is a new type of recursive proof system that uses folding scheme as its core building block. This brilliant idea of folding relations can significantly reduce recursion overhead. In this paper, we study some issues related to Nova's soundness proof that relies on the soundness of the folding scheme in a recursive manner.

First, its proof strategy, due to its recursive nature, inevitably expands the running time of the recursive extractor polynomially for each additional recursive step. This constrains Nova's soundness model to have only logarithmically bounded recursive steps, and so the soundness proof in this limited model does not guarantee the soundness for linear rounds in the security parameter, e.g., 128 rounds for 128 bit security. On the other hand, there are no known attacks on arbitrary depth recursion of Nova, leaving a gap between theoretical security guarantees and real-world attacks. We aim to bridge this gap in two opposite directions. In a negative direction, we present a recursive proof system that is unforgeable in a log-round model but forgeable if used in linear rounds. This shows that the soundness proof in the log-round model might tell nothing about real-world uses that require linearly long rounds. In a positive direction, we show that when Nova uses a specific group-based folding scheme, its knowledge soundness over polynomial rounds can be proven in the algebraic group model with our refinements. To the best of our knowledge, this is the first result to show Nova's polynomial rounds soundness.

Second, the folding scheme is converted non-interactively via the Fiat-Shamir transformation and then arithmetized into R1CS. Therefore, the soundness of Nova using the non-interactive folding scheme essentially relies on the heuristic random oracle instantiation in the standard model. In our new soundness proof for Nova in the algebraic group model, we replace this heuristics with a cryptographic hash function with special property. We view this hash function as an independent object of interest and expect it to help further understand the soundness of Nova.

1 Introduction

Incrementally Verifiable Computation (IVC) [49] and its generalization, Proof-Carrying Data (PCD) [23] are cryptographic primitives that facilitate the generation of proofs that convince the accurate execution of lengthy computations.

* corresponding author

These proofs enable efficient verification by a verifier for any prefix of the computation. IVC schemes find applications in diverse domains, such as verifiable delay functions (VDF) [7, 36], succinct blockchains [12, 24, 11, 34], and verifiable state machines [41].

VDF schemes are one of the key tools for Ethereum's consensus protocols, and several studies have incorporated the IVC scheme into VDF [36]. VDF involves recursive computation, and IVC enables efficient verification even when the computation is computationally expensive.

There are also IVC-based succinct blockchain projects [12, 24, 11]. The IVC scheme allows for avoiding the need to download the full history for verification. Using the current state with IVC proof, a node can verify the validity of the current state and all previous states. If the IVC scheme is applied to Ethereum, which has a market capitalization of approximately hundreds of billions dollars and provides approximately 13.4 seconds for block generation times [26], it would require approximately $6,000$ recursive computations per day. Therefore, the IVC scheme for these applications should provide an appropriate level of security for large recursive steps.

Although many proposals for IVC/PCD schemes [6, 13, 17, 16, 40] offer provable security, their knowledge soundness is proven only in a limited model with at most $O(\log \lambda)$ recursive rounds, where λ is the security parameter. This is because the common proof strategy applied in those proposals is to construct a recursive extractor that blows up polynomially for each additional recursive step. Thus, recursion can be performed only for $O(\log \lambda)$ rounds before the extractor's running time becomes super-polynomial in λ. In fact, there are PCD schemes achieving polynomially-long chains [23, 19], but those require additional strong assumption such as hardware tokens or are relatively impractical compared to practical constructions such as Nova [40], a new type of recursive proof system.

Nova uses a folding scheme as its core building block. This brilliant idea of folding relations can significantly reduce the recursion overhead. Nova's soundness proof follows the common proof strategy of using a general recursive technique, and thus is also proven in the aforementioned limited model with $O(\lambda)$ rounds. Therefore, Nova's soundness proof does not guarantee the soundness for linear rounds in the security parameter, for example, 128 rounds for 128 bit security, which is too short to be used in various aforementioned applications. This limitation of the current IVC model has been pointed out in several literature [40, 45]. Nevertheless, there are no known attacks on arbitrary depth recursion, leaving a gap between theoretical security guarantees and real-world attacks.

Our Contribution. Our contribution is twofold. First, we separate the gap between theoretical security guarantees obtainable from the limited IVC model with $O(\log \lambda)$ recursive rounds and the knowledge soundness in the IVC model without log-round bounds. To this end, we present a variant of Nova, called Ephemeral-Nova, that satisfies the knowledge soundness in the limited IVC model with $O(\log \lambda)$ recursive rounds, but is forgeable in the IVC model with a linear round in λ. Therefore, Ephemeral-Nova shows the need for a new proof strategy for knowledge soundness in the IVC model without a log-round bound.

The second contribution is a new security proof for the knowledge soundness proof for Nova from a group-based folding scheme. In fact, the folding scheme proposed in the Nova paper [40] is a group-based construction; thus, ours is a new security proof for the concrete scheme given in [40]. To the best of our knowledge, our security proof is the first result that demonstrates the knowledge soundness of Nova in polynomial rounds and partially explains why there are no known attacks against Nova on arbitrary depth recursion. Furthermore, in our new soundness proof, we remove the heuristics used in the original proof; the non-interactive folding scheme is obtained by applying the Fiat-Shamir transformation to its interactive version. Then, it is arithmetized into R1CS, so that the random oracle instantiation is public to the adversary. In fact, many IVC scheme using the Fiat-Shamir transformation relies on a similar heuristic assumption. We introduce a new property for cryptographic hash function that is about computationally hardness, like preimage-resistance and collision-resistance. Then, we use it for our new soundness proof for Nova in the algebraic group model, without relying on random oracles.

Our Idea for Designing Ephemeral-Nova. Together with an execution function $F : \mathcal{Z} \times \mathcal{W} \to \mathcal{Z}$ and two values $z_0, z_n \in \mathcal{Z}$, a prover of an IVC scheme generates a succinct proof that proves the knowledge of $\omega_0, \ldots, \omega_{n-1}$ that satisfy the relations $F(z_{i-1}, \omega_{i-1}) = z_i$ for $i = 1, \ldots, n$. Nova's idea for the new IVC design is to use a folding scheme that can fold two instance-witness pairs into one pair and apply the folding scheme to fold the instances for the augmented execution function F'. Here, the augmented function F' includes several necessary checks and computations, such as the execution of F and the folding procedure.

Although it is necessary for the augmented function F' to include the necessary procedures for soundness, such as the execution of F, we found that adding some redundant precedure may not harm the knowledge soundness of the IVC scheme. From this observation, one can try to inject some trigger into F' such that it only becomes activated after a sufficiently large number of rounds. For this purpose, such a trigger should be controllable for the timing of activation and also deterministic because the execution of F' should be arithmetized into R1CS. For Ephemeral-Nova, we found an appropriate trigger that can be summarized as the following recursive sequence:

$$Y_{n+1} := Y_n^{2\alpha} \cdot A_n \pmod q \text{ and } Y_0 := 1,$$

where q is a prime number of the form $2^k\alpha+1$ for $k \geq \lambda$ and odd arbitrary integer α. Suppose that each A_n is either 1 or chosen from a uniform distribution. If $n < k$, then $Y_{n+1} = 1$ is almost equivalent to the case in which all $A_0, \ldots, A_n$ are ones. This equivalence is maintained until n is sufficiently smaller than k, but is suddenly broken if n exceeds k. This sequence contains a sudden transition in the equivalence, the timing of which can be controlled by selecting q, and the uniform distribution of A_n can be replaced with a deterministic procedure such as a cryptographic hash function. Using this special sequence, we can construct Ephemeral-Nova whose behavior is almost equivalent to the original Nova before the linear round and satisfies the knowledge soundness in the constrained IVC model with a log-round bound, but is forgeable after the linear round due to the activated trigger.

The design of Ephemeral-Nova allows us to find that unnecessary steps in F' may cause a problem that cannot be captured by a general recursive proof strategy. Therefore, new knowledge soundness proof strategies are needed that can investigate all unexpected effects, including the above trigger.

Our Idea for New Knowledge Soundness Proof for Polynomial Rounds. Nova's soundness proof relies on the soundness of the underlying folding scheme and uses a recursive proof strategy to extract the witness ω_i's in reverse order. Let $\mathcal{E}_i$ be an extractor to extract ω_i, $\tilde{\mathcal{A}}_i$ be an adversary for the folding scheme, and $\tilde{\mathcal{E}}_i$ be an extractor for the folding scheme. Then, the recursive proof strategy leads an inequality between the running time: $\mathtt{time}(\mathcal{E}_i) > \mathtt{time}(\tilde{\mathcal{E}}_i) + \mathtt{time}(\tilde{\mathcal{A}}_i) > 2 \cdot \mathtt{time}(\mathcal{E}_{i+1})$, where the right inequality holds if $\mathtt{time}(\tilde{\mathcal{E}}_i) > \mathtt{time}(\tilde{\mathcal{A}}_i)$. Therefore, the running time required to extract all ω_i increases exponentially in the final number of rounds.

To avoid recursive blowup, instead of relying on the soundness of the folding scheme, we directly prove the soundness of IVC scheme. This requires a direct procedure to extract all ω_i from the attacker's output $(F, (z_0, z, \mathit{\Pi}))$ only, where F is an execution function, z_0 is an initial input of F, z is the final output of F, and $\mathit{\Pi}$ is a valid IVC proof. Indeed, the adversary's output is too limited to extract all intermediate ω_i, without an additional resource such as a folding extractor. Therefore, we move to an ideal model to see a partial history of group-related operations performed by the adversary until the final result is output, where the underlying folding scheme is group-based. There are two well-established ideal models for handling group operations: the generic group model (GGM) [44, 48, 43] and algebraic group model (AGM) [28]. GGM is devised to demonstrate the hardness of group-based problems or the security of crypto schemes against attackers who are constrained not to use group descriptions. In the AGM, all group elements that the attack algorithm outputs are derived from known group elements via group operations. In order to analyze the security of Nova IVC scheme, both GGM and AGM have limitations. The GGM has the advantage of tracking the history of group operations because of its interactive feature. In the Nova IVC scheme, however, the folding verifier is arithmetized into R1CS so that group operations should be instantiated in R1CS, which is not allowed in the GGM. In fact, a similar situation occurs when we use the random oracle model in the analysis of non-interactive folding scheme. That is, the cryptographic hash functions are modeled as the random oracle, but the hash function should be instantiated in R1CS when the folding verifier is arithmetized into R1CS. Heuristically, one might assume that these are securely possible. We avoid these heuristics as much as possible. (We will revisit the random oracle model later.) In the AGM, the adversary should output a representation vector whenever a group element is output. The AGM has another limitations. Recently, Zhang, Zhou and Katz [51] and Zhandry [50] showed that there are computational games that security analysis against algebraic algorithms guarantees neither the non-existence of general attacks nor the non-existence of standard model attacks. We take close look at such computational games and refine the AGM to rule out such exceptional situations. In addition, we further refine the AGM to further cover a broader range of algorithms by allowing to use multiple encodings for each group element. This simple extension enables to get more information from the algebraic adversary in some cases. For example, using multiple encodings of group elements, a representation vector the algebraic algorithm outputted might have a (different encoding of) group element, then we can ask the algebraic algorithm to output the corresponding representation at the same time. This can be repeatedly done until a representation vector does not contain any group element. In fact, unlike the GGM, the original AGM does not allow for a challenger to see the history of group element processing of algebraic algorithm. Our refinement partially provides for the challenger with opportunity to view the history of group element processing if group elements are embedded into exponents sequentially.

There are studies [20, 19] that aim at removing the heuristic instantiations of random oracles by introducing new variants of random oracles. We propose a different approach for avoiding heuristic analysis because we do not want to change the Nova IVC construction but rather provide a new soundness analysis. To this end, we propose a new plausible property of cryptographic hash functions such as SHA-256 that is sufficient for proving the knowledge soundness in the AGM. Note that the new property of hash function we introduce is an intractability property, like preimgage-resistance and collision-resistance. That is, this property of hash function cannot solely replace the random oracles because it cannot replace such a power of the random oracle to extract witness by rewinding algorithms. However, we use this property of hash functions with our AGM refinement; Our AGM refinement is useful to extract something the adversary used, and then this property of hash function can be used to show the extracted ones satisfy some relations so that eventually are witness of R1CS.

Additional Related Works. A well-known approach for IVC is to recursively utilize succinct non-interactive arguments of knowledge (SNARKs) [30, 31] for arithmetic circuits. In this approach [4], at each incremental step i, the prover generates a SNARK proving the correct execution of F to the output of step i and that the SNARK verifier, represented as a circuit, has accepted the SNARK for step $i - 1$. However, SNARK-based approaches are considered impractical because they require a cycle of pairing friendly elliptic curves. Furthermore, this approach requires a trusted setup that inherits from SNARKs. To address this issue, there are alternative approaches using NARKs [13, 17, 16], by deferring expensive verification circuit per each step.

Organization. The next section describes Nova IVC and folding method, which is the core building block of Nova. In Section 3, we propose a new IVC scheme called Ephemeral-Nova that has knowledge soundness in log-bounded rounds but is forgeable in linear rounds. In Section 4, we review idealized models for group-based systems and refine to adapt to the situation for Nova IVC. In Section 5, we introduce a new property of hash function and show how to use it in the AGM to replace with the random oracles. In Section 6, we present a new knowledge soundness proof for Nova from group-based folding scheme in the refined AGM. Finally, we provide concluding remarks in Section 7.

2 IVC from Folding Scheme

Notation. We first define notations used in the paper. $[m]$ denotes a set of integers from 1 to m, $\{1, \cdots, m\}$. Let $\mathbb{Z}_p$ be the ring of integers modulo p. Uniform sampling is denoted by $\overset{\$}{\leftarrow}$. For instance, $a \overset{\$}{\leftarrow} \mathbb{Z}_p$ indicates that a is uniformly chosen from $\mathbb{Z}_p$. We use bold font to represent vectors. For $\boldsymbol{a} = (a_1, \ldots a_\ell)$ and $\boldsymbol{b} = (b_1, \ldots, b_\ell) \in \mathbb{Z}_p^\ell$, three binary operations, concatenation, hadamard product and inner product, are represented by $\boldsymbol{a} \| \boldsymbol{b} = (a_1, \ldots, a_\ell, b_1, \ldots, b_\ell)$, $\boldsymbol{a} \circ \boldsymbol{b} = (a_1 b_1, \ldots, a_\ell b_\ell)$ and $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i=1}^{\ell} a_i b_i$, respectively. hash denote a cryptographic hash function whose range will be specified from the context.

Definition 1 (Commitment Scheme). *A commitment scheme is defined by two PPT algorithms: the setup algorithm* Setup *and commitment algorithm* Com*. Let* M*,* R*, and* C *be message space, random space, and commitment space respectively.* Setup *and* Com *are defined by:*

- $\mathsf{Setup}(1^\lambda, \ell) \to \mathsf{ck}$ *: On input security parameter* λ *and dimension of message space* ℓ*, sample commitment key* ck
- $\mathsf{Com}(\mathsf{ck}, m; r) \to C$ *: Take commitment key* ck*, message* $m \in \mathsf{M}$*, and randomness* $r \in \mathsf{R}$*, output commitment* $C \in \mathsf{C}$

We call $(\mathsf{Setup}, \mathsf{Com})$ *a commitment scheme if the following two properties hold:*
[`Binding`*]: For any expected PPT adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c}\mathsf{Com}(\mathsf{ck}, m_0; r_0) = \mathsf{Com}(\mathsf{ck}, m_1; r_1), \\ \wedge\ m_0 \neq m_1\end{array}\middle|\begin{array}{c}\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, \ell), \\ (m_0, r_0, m_1, r_1) \leftarrow \mathcal{A}(\mathsf{ck})\end{array}\right] \leq \mathsf{negl}(\lambda)$$

[`Hiding`*]: For any expected PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

$$\left|\Pr\left[\mathsf{b} = \mathsf{b}' \middle| \begin{array}{c}\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, \ell), \\ (m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{ck}), \\ \mathsf{b} \overset{\$}{\leftarrow} \{0,1\}, r \overset{\$}{\leftarrow} \mathcal{R}, C \leftarrow \mathsf{Com}(\mathsf{ck}, m_\mathsf{b}; r), \\ \mathsf{b}' \leftarrow \mathcal{A}_2(\mathsf{ck}, C, \mathsf{state}),\end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

Let M*,* R*, and* C *be efficiently computable (additive) groups. Then, We call a commitment scheme* $(\mathsf{Setup}, \mathsf{Com})$ homomorphic *if the* $(\mathsf{Setup}, \mathsf{Com})$ *satisfying the following homomorphic property.*
[`Homomorphic`*]: For any commitment key* $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda, N)$ *and pairs of message-randomness* $(m_0, r_0), (m_1, r_1) \in \mathsf{M} \times \mathsf{R}$*, the following equation holds:*

$$\mathsf{Com}(\mathsf{ck}, m_0; r_0) + \mathsf{Com}(\mathsf{ck}, m_1; r_1) = \mathsf{Com}(\mathsf{ck}, m_0 + m_1; r_0 + r_1)$$

2.1 Definitions of IVC and (Refined) Folding Scheme

Definition 2 (IVC). *An incrementally verifiable computation (IVC) scheme is defined by three PPT algorithms the generator $\mathcal{G}$, the prover $\mathcal{P}$, the verifier $\mathcal{V}$, and one deterministic polynomial time algorithm the encoder $\mathcal{K}$. We say that an IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ satisfies perfect completeness if for any PPT adversary $\mathcal{A}$*

$$\Pr\left[\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1 \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ F, (i, z_0, z_{i-1}, \omega_{i-1}, \Pi_{i-1}) \leftarrow \mathcal{A}(\mathsf{pp}), \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F), \\ z_i = F(z_{i-1}, \omega_{i-1}), \\ \mathcal{V}(\mathsf{vk}, i-1, z_0, z_{i-1}, \Pi_{i-1}) = 1, \\ \Pi_i \leftarrow \mathcal{P}(\mathsf{pk}, i, z_0, z_{i-1}, \omega_{i-1}, \Pi_{i-1}) \end{array}\right] = 1$$

where F is a polynomially efficient computable function. We say that an IVC scheme satisfies knowledge-soundness if for arbitrary polynomial $n = poly(\lambda)$, and expected polynomial time adversaries $\mathcal{P}^*$, there exists expected polynomial-time extractor $\mathcal{E}$ such that for any input randomness ρ

$$\Pr\left[\begin{array}{c} z_n \neq z, \\ \textit{where } z_i \leftarrow F(z_{i-1}, \omega_{i-1}) \textit{ for } i \in [n], \\ \wedge \\ \mathcal{V}(\mathsf{vk}, n, z_0, z, \Pi) = 1 \end{array} \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ F, (z_0, z, \Pi) \leftarrow \mathcal{P}^*(\mathsf{pp}; \rho), \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F), \\ (\omega_i)_{i=0}^{n-1} \leftarrow \mathcal{E}(\mathsf{pp}, z_0, z; \rho) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Finally, we say that an IVC scheme satisfies succinctness if the size of the IVC proof Π is independent from the number of applications n.

We additionally define a weaker knowledge-soundness definition by restricting the number of applications n to be bounded by a logarithm function and call it the *log-bounded round IVC*, which covers the definition used in not only Kothapalli et al. [40] but also all other works using general recursive techniques [13, 17, 22, 16, 9, 38, 39, 15, 45].

To define a folding scheme, we consider a special relation $\mathcal{R}$ over tuples consisting of public parameters pp_{FS}, structure s, instance u, and witness v. We use the notation $\mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$ to denote the subset $(\mathsf{pp}_{FS}, \mathsf{s}, \cdot, \cdot) \subset \mathcal{R}$ if pp_{FS} and s are fixed. Informally, the folding scheme has, beyond two interactive prover P and verifier V, additional algorithms G and K that specify the first two terms of $\mathcal{R}$, pp_{FS} and s. After fixing pp_{FS} and s, a folding scheme allows two instance-witness pairs $(\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$ to be folded into one pair $(\mathsf{u}, \mathsf{v}) \in \mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$ and the soundness of the folding scheme informally says that if two instances u_1 and u_2 are folded and the folded instance-witness pair (u, v) is included in $\mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$, then there are valid witness v_1 and v_2 satisfying $(\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$. The formal definition of folding scheme is given below.

Definition 3 ((Refined) Folding Scheme). *Consider a relation $\mathcal{R}$ over public parameters, structure, instance, and witness tuples. A folding scheme for $\mathcal{R}$ consists of three PPT algorithms, a generator G, a prover P and a verifier V, and a deterministic key generation algorithm K, all defined as follows.*

- $\mathsf{G}(1^\lambda, N) \to \mathsf{pp}_{FS}$ *: On input security parameter λ and the maximum size of common structure N, samples public parameters pp_{FS}*
- $\mathsf{K}(\mathsf{pp}_{FS}, \mathsf{s}) \to \mathsf{pk}_{FS}$*: On input pp_{FS} and a common structure s, of size N, between instances to be folded, outputs a prover key pk_{FS}.*
- $\mathsf{P}(\mathsf{pk}_{FS}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \to (\mathsf{u}, \mathsf{v})$*: On input two instance-witness pairs $(\mathsf{u}_1, \mathsf{v}_1)$ and $(\mathsf{u}_2, \mathsf{v}_2)$, outputs a new instance-witness pair (u, v) of the same size and folding proof Π to allow the verifier update new instance.*
- $\mathsf{V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, \Pi) \to \mathsf{u}$*: On input two instances u_1 and u_2, outputs a new instance u.*

Although the final outputs of P and V are defined in the above description, both are interactive algorithms and thus the interactive procedure and the corresponding transcript are, respectively, denoted as follows.

$$(\mathsf{u}, \mathsf{v}) \leftarrow \langle \mathsf{P}(\mathsf{pk}_{FS}, \mathsf{v}_1, \mathsf{v}_2), \mathsf{V}(\mathsf{pp}_{FS}) \rangle(\mathsf{u}_1, \mathsf{u}_2), \;\; tr = \langle \mathsf{P}(\mathsf{pk}_{FS}, \mathsf{v}_1, \mathsf{v}_2), \mathsf{V}(\mathsf{pp}_{FS}) \rangle(\mathsf{u}_1, \mathsf{u}_2)$$

A folding scheme for $\mathcal{R}$ satisfies the following requirements.
1. Perfect Completeness: For all PPT adversaries $\mathcal{A}$, we have that

$$\Pr\left[(\mathsf{pp}_{FS}, \mathsf{s}, \mathsf{u}, \mathsf{v}) \in \mathcal{R} \;\middle|\; \begin{array}{c} \mathsf{pp}_{FS} \leftarrow \mathsf{G}(1^\lambda, N), \\ (\mathsf{s}, (\mathsf{u}_1, \mathsf{u}_2), (\mathsf{v}_1, \mathsf{v}_2)) \leftarrow \mathcal{A}(\mathsf{pp}_{FS}), \\ (\mathsf{pp}_{FS}, \mathsf{s}, \mathsf{u}_1, \mathsf{v}_1), (\mathsf{pp}_{FS}, \mathsf{s}, \mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}, \\ \mathsf{pk}_{FS} \leftarrow \mathsf{K}(\mathsf{pp}_{FS}, \mathsf{s}), \\ (\mathsf{u}, \mathsf{v}) \leftarrow \langle \mathsf{P}(\mathsf{pk}_{FS}, \mathsf{v}_1, \mathsf{v}_2), \mathsf{V}(\mathsf{pp}_{FS}) \rangle(\mathsf{u}_1, \mathsf{u}_2) \end{array} \right] = 1.$$

2.Knowledge Soundness : For any expected PPT adversary $\tilde{\mathcal{A}} = (\mathcal{A}, \mathsf{P}^*)$ there is an expected polynomial-time extractor $\mathcal{E}$ such that over all randomness ρ

$$\Pr\left[\begin{array}{c} (\mathsf{pp}_{FS}, \mathsf{s}, \mathsf{u}_1, \mathsf{v}_1) \in \mathcal{R}, \\ (\mathsf{pp}_{FS}, \mathsf{s}, \mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R} \end{array} \;\middle|\; \begin{array}{c} \mathsf{pp}_{FS} \leftarrow \mathsf{G}(1^\lambda, N), \\ (\mathsf{s}, (\mathsf{u}_1, \mathsf{u}_2)) \leftarrow \mathcal{A}(\mathsf{pp}_{FS}, \rho), \\ (\mathsf{v}_1, \mathsf{v}_2) \leftarrow \mathcal{E}(\mathsf{pp}_{FS}, \rho) \end{array} \right] \approx$$
$$\Pr\left[(\mathsf{pp}_{FS}, \mathsf{s}, \mathsf{u}, \mathsf{v}) \in \mathcal{R} \;\middle|\; \begin{array}{c} \mathsf{pp}_{FS} \leftarrow \mathsf{G}(1^\lambda), \\ (\mathsf{s}, (\mathsf{u}_1, \mathsf{u}_2)) \leftarrow \mathcal{A}(\mathsf{pp}_{FS}, \rho), \\ \mathsf{pk}_{FS} \leftarrow \mathsf{K}(\mathsf{pp}_{FS}, \mathsf{s}), \\ (\mathsf{u}, \mathsf{v}) \leftarrow \langle \mathsf{P}^*(\mathsf{pk}_{FS}, \rho), \mathsf{V}(\mathsf{pp}_{FS}) \rangle(\mathsf{u}_1, \mathsf{u}_2) \end{array} \right]$$

Definition 4 (Public Coin). *A folding scheme $(\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ is called public coin if all the messages sent from V to P are sampled from a uniform distribution.*

Definitional Refinement. In our refined definition of folding scheme, the verifier V takes pp_{FS} as input, unlike the prover P which takes pk_{FS} as input. In the original definition of folding scheme [40], V also takes vk_{FS} as input, where vk_{FS} is generated by both pp_{FS} and s. Our definition is a special case of the original definition since vk_{FS} can be set by pp_{FS}. We argue that our refinement is necessary if the folding scheme is used in the IVC design. Looking at the use of folding scheme in the IVC design in [40], the folding verifier should be a part of the augmented function F', which is arithmetized to the (committed relaxed)

R1CS. That is, the description of V should be contained in s and thus V should not take s as input to avoid a circular contradiction. In particular, the concrete group-based construction of folding scheme in [40] satisfies our refined definition because its process does not require s.

Committed Relaxed R1CS. The committed relaxed R1CS is a variant of the R1CS constraints system, which is widely used in proof system [47, 18, 21, 16]. In particular, the committed relaxed R1CS is a public parameter-dependent relation defined over public parameters. Let us explain the committed relaxed R1CS in terms of folding scheme. The public parameter generator of the folding scheme G takes the size parameter N as the input. We specify N to have two positive integers m and ℓ with $\ell + 1 < m$. G outputs public parameter pp_{FS} that consists of commitment keys of homomorphic commitment scheme Com for committing vectors over a finite field $\mathbb{Z}_p$. More precisely, $\mathsf{pp}_{FS} = (\mathsf{pp}_{\boldsymbol{w}}, \mathsf{pp}_{\boldsymbol{e}})$, which are two public parameters of Com with dimensions m and $m - \ell - 1$, respectively. The structure s indicates the R1CS parameter matrices $A, B, C \in \mathbb{Z}_p^{m\times m}$, where there are at most $\Omega(m)$ non-zero entries in each matrix and they specify the R1CS relation $A\boldsymbol{x} \circ B\boldsymbol{x} = C\boldsymbol{x}$. Note that the dimensions of the matrices are already specified in N.

The committed relaxed R1CS relation is the relation with parameter $\mathsf{pp}_{FS} = (\mathsf{pp}_{\boldsymbol{w}}, \mathsf{pp}_{\boldsymbol{e}})$ and structure $\mathsf{s} = (A, B, C)$ defined by

$$\mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}} = \left\{ \big((E, W, s, \mathsf{x}); (\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}})\big) : \begin{array}{r} E = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{e}}, \boldsymbol{e}; r_{\boldsymbol{e}}) \\ W = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{w}}, \boldsymbol{w}; r_{\boldsymbol{w}}) \\ \boldsymbol{z} = (\boldsymbol{w}, \mathsf{x}, s) \\ A\boldsymbol{z} \circ B\boldsymbol{z} = sC\boldsymbol{z} + \boldsymbol{e} \end{array} \right\}, \quad (1)$$

where x is public inputs and outputs.

Note that if one adds conditions $\boldsymbol{e} = \mathbf{0}$ and $s = 1$ in the above relation, the resulting relation becomes equivalent to the R1CS relation specified by the structure s.[1]

Non-Interactive Folding Scheme. Given a public-coin interactive folding scheme, it can be transformed to an non-interactive folding scheme, defined below, in the random oracle model via the Fiat-Shamir transform [27].

Definition 5 (Non-Interactive). *We say that a folding scheme (*G,K,P,V*) is non-interactive if the interaction between* P *and* V *consists of a single message* T *from* P *to* V*. To clearly indicate the single message interaction, the input and output of* P *and* V *can be re-written as* $\mathsf{P}(\mathsf{pk}_{FS}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \to (\mathsf{u}, \mathsf{v})$*,* T *and* $\mathsf{V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, T) \to \mathsf{u}$*.*

In fact, the folding prover and verifier are implemented in the design of Nova IVC; therefore, we must heuristically instantiate the random oracle using

[1] In [40], the alphabet u is used instead of s in this paper. We changed it to avoid the confusion since u_i are used to denote an instance of the relation. Similarly, we use v to denote witness.

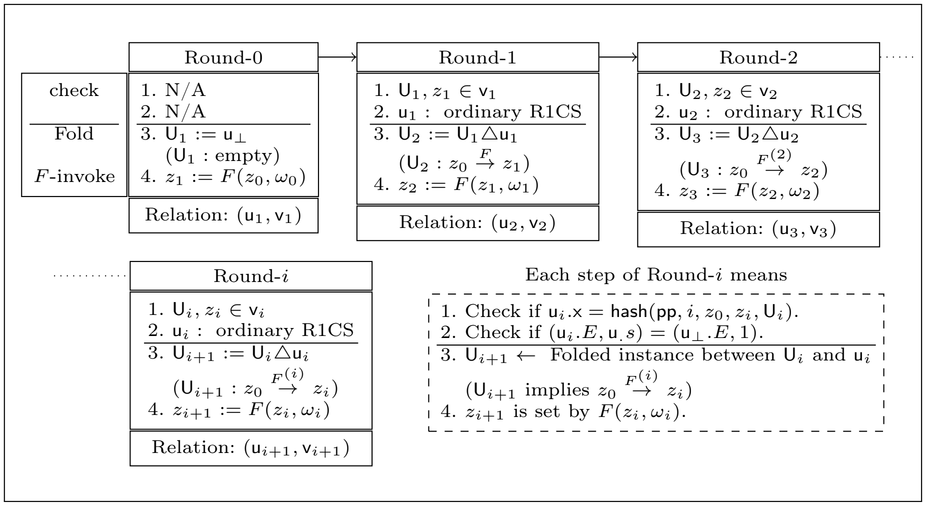

check
Fold
F-invoke
Round-0
1. N/A
2. N/A
3. $\mathsf{U}_1 := \mathsf{u}_\perp$
$(\mathsf{U}_1$: empty$)$
4. $z_1 := F(z_0, \omega_0)$
Relation: $(\mathsf{u}_1, \mathsf{v}_1)$
Round-1
1. $\mathsf{U}_1, z_1 \in \mathsf{v}_1$
2. u_1 : ordinary R1CS
3. $\mathsf{U}_2 := \mathsf{U}_1 \triangle \mathsf{u}_1$
$(\mathsf{U}_2 : z_0 \overset{F}{\to} z_1)$
4. $z_2 := F(z_1, \omega_1)$
Relation: $(\mathsf{u}_2, \mathsf{v}_2)$
Round-2
1. $\mathsf{U}_2, z_2 \in \mathsf{v}_2$
2. u_2 : ordinary R1CS
3. $\mathsf{U}_3 := \mathsf{U}_2 \triangle \mathsf{u}_2$
$(\mathsf{U}_3 : z_0 \overset{F^{(2)}}{\to} z_2)$
4. $z_3 := F(z_2, \omega_2)$
Relation: $(\mathsf{u}_3, \mathsf{v}_3)$
Round-i
1. $\mathsf{U}_i, z_i \in \mathsf{v}_i$
2. u_i : ordinary R1CS
3. $\mathsf{U}_{i+1} := \mathsf{U}_i \triangle \mathsf{u}_i$
$(\mathsf{U}_{i+1} : z_0 \overset{F^{(i)}}{\to} z_i)$
4. $z_{i+1} := F(z_i, \omega_i)$
Relation: $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1})$
Each step of Round-i means
1. Check if $\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i)$.
2. Check if $(\mathsf{u}_i.E, \mathsf{u}_. s) = (\mathsf{u}_\perp.E, 1)$.
3. $\mathsf{U}_{i+1} \leftarrow$ Folded instance between U_i and u_i
$(\mathsf{U}_{i+1}$ implies $z_0 \overset{F^{(i)}}{\to} z_i)$
4. z_{i+1} is set by $F(z_i, \omega_i)$.

Fig. 1. Informal Description of Relation (u, v) for Each Round of Nova

a cryptographic hash function. Therefore, we can only heuristically argue the security of the resulting non-interactive folding scheme in the standard model. To the best of our knowledge, all existing IVC proposals in the standard model rely on the same heuristics that require instantiating the random oracle with a cryptographic hash function [40, 38, 39, 15, 45].

2.2 Nova: IVC from Folding Scheme

Given a function F, an IVC scheme iteratively invokes the computation of F for each round. Nova [40] is an IVC scheme built from a folding scheme such that the computation in each round is an augmented function F' that not only invokes F but also folds two committed relaxed R1CS instances, where F' is represented by the committed relaxed R1CS.

An informal description of computation in each round is given in Fig. 1, where hash is a cryptographic hash function and $(\mathsf{u}_\perp, \mathsf{v}_\perp)$ is a trivial instance-witness pair such that $\mathsf{v}_\perp$ is set by zeros. In addition, we define the trivial proof $\Pi_0 = (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp, \mathsf{v}_\perp)$, which consists of two trivial instance-witness pairs.

Let $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ be the non-interactive folding scheme for the committed relaxed R1CS of F'. The formal descriptions of the augmented function F' and Nova from NIFS are, respectively, provided in Fig. 2 and Fig. 3. Here, trace is a compiler that coverts from an execution of F' on non-deterministic advice $(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T)$ to the corresponding committed relaxed R1CS instance-witness pair $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1})$, where the advice is a part of v_{i+1} and the output hash value of F' is only the public IO of u_{i+1}, that is, $\mathsf{u}_{i+1}.\mathsf{x}$.

Theorem 1 (Nova-IVC [40]). *If the non-interactive folding scheme* NIFS *satisfy perfect completeness and knowledge soundness, then Nova in the Fig. 3 is*

```
$\underline{F'(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T) \to \mathsf{x}}$:
  If $i$ is 0, output $\mathsf{hash}(\mathsf{pp}, 1, z_0, F(z_0, \omega_i), u_\perp)$;
  otherwise,
    1. check that $\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i)$, where $\mathsf{u}_i.\mathsf{x}$ is the public IO of $\mathsf{u}_i$
    2. check that $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\perp.E, 1)$
    3. compute $\mathsf{U}_{i+1} \leftarrow \mathsf{NIFS.V}(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, T)$, and
    4. output $\mathsf{hash}(\mathsf{pp}, i+1, z_0, F(z_i, \omega_i), \mathsf{U}_{i+1})$.
```

Fig. 2. Augmented Function F'

```
$\underline{\mathcal{G}(1^\lambda) \to \mathsf{pp}}$: Output $\mathsf{pp} \leftarrow \mathsf{NIFS.G}(1^\lambda, N)$.

$\underline{\mathcal{K}(\mathsf{pp}, F) \to (\mathsf{pk}, \mathsf{vk})}$:  1. Run $\mathsf{pk}_{FS} \leftarrow \mathsf{NIFS.K}(\mathsf{pp}, \mathsf{s}_{F'})$
                          2. Output $(\mathsf{pk}, \mathsf{vk}) \leftarrow ((F, \mathsf{pk}_{FS}), (F, \mathsf{pp}))$

$\underline{\mathcal{P}(\mathsf{pk}, (i, z_0, z_i), \omega_i, \Pi_i) \to \Pi_{i+1}}$:
  Parse $\Pi_i$ as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ and then
    1. if $i$ is 0, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp.E)$;
       otherwise, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow \mathsf{NIFS.P}(\mathsf{pk}, (\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$
    2. compute $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1}) \leftarrow \mathsf{trace}(F', (\mathsf{vk}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T))$, and
    3. output $\Pi_{i+1} \leftarrow ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

$\underline{\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \Pi_i) \to \{0, 1\}}$:
  If $i$ is 0, check that $z_0 = z_i$;
  otherwise,
    1. parse $\Pi_i$ as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$,
    2. check if $\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i)$,
    3. check if $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\perp.E, 1)$, and
    4. check if $(\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, the committed relaxed R1CS induced by $F'$.
```

Fig. 3. Nova IVC

a log-bounded round IVC scheme satisfying perfect completeness and knowledge soundness.

3 Ephemeral-Nova: A New Log-bounded round IVC

This section explores whether the security proof for log-bounded round IVC scheme can provide an appropriate level of soundness guarantees for a linear number of rounds. In particular, we demonstrate that at least not all log-bounded round IVC schemes are knowledge-sound for a linear number of recursive rounds. To this end, we design a variant of Nova, called Ephemeral-Nova, that satisfies the knowledge-soundness definition of a log-bounded round IVC scheme but is forgeable when used more than a linearly large number of recursive rounds.

Our Idea for Ephemeral-Nova. Basically, the Ephemeral-IVC scheme should be knowledge-sound in the log-bounded round model, and thus, we begin by looking at the original proof of knowledge-soundness of Nova. We first notice that the polynomial time extractor in the original proof of knowledge-soundness can extract the witness in the last $O(\log \lambda)$ number of rounds, where λ is the security parameter, because the running time of the extractor blows up polynomially for each additional recursion round. From this observation, we find that to design a *linearly-faulty-and-logarithmically-provable* scheme, the verification procedure of the Ephemeral-IVC scheme should be in such a way of

- `[Faulty]` pardon for misbehavior before last log number of rounds, but
- `[Provable]` correctly checking the validity of the last log number of rounds.

Furthermore, it should be

- `[Compile]` deterministic to be compiled into the committed relaxed R1CS.

Designing an IVC satisfying the above requirements is somewhat challenging because the timing of log number of rounds depends on the security parameter. Therefore, we need to devise a deterministic process of gradual change of (un)soundness in the security parameter. To this end, we first devise a recursive sequence with the above three features as follows.

$$Y_{n+1} := Y_n^{2\alpha} \cdot A_n \pmod q \text{ and } Y_0 := 1, \tag{2}$$

where q is a prime number of the form $\alpha \cdot 2^k + 1$ for some $k \geq \lambda$ and odd integer α, which are at most λ-bit, and A_n, which is selected from one of two distributions, either a constant 1 or uniform distribution on $\mathbb{Z}_q$. Suppose that for the values A_i, we regard 1 as normal and use Y_i for verification of the normality of all $A_0, \ldots, A_{i-1}$. Solving the recurrences of Eq. (2), we obtain

$$Y_{n+1} = \prod_{i=0}^{n} A_i^{(2\alpha)^{n-i}} \pmod q. \tag{3}$$

For time step $n = O(\log \lambda)$, if all previous A_i $(i = 0, .., n)$ are normal, then we have $Y_{n+1} = 1$. If at least one A_i is abnormal, then $Y_{n+1} \neq 1$ except for the negligible probability in λ since A_i is uniformly distributed, $q > 2^\lambda$, and $n = O(\log \lambda)$. Therefore, checking $Y_{n+1} = 1$ is a good verification procedure for normality of all previous A_i $(i = 0, .., n)$. However, when time step n becomes large enough (e.g., $n \geq k \geq \lambda$), $Y_{n+1} = 1$ does not guarantee the normality of all previous A_i. This is due to the shape of the prime number q and the Fermat's Little Theorem as follows.

$$\begin{aligned} Y_{n+1} &= \prod_{i=0}^{n} A_i^{(2\alpha)^{n-i}} \pmod q \\ &= \prod_{i=n-k+1}^{n} A_i^{(2\alpha)^{n-i}} \pmod q \text{ (by Fermat's Little Theorem)} \end{aligned}$$

Therefore, checking $Y_{n+1} = 1$ is a good verification procedure for normality of only the last $k-1$ values $A_n, \ldots, A_{n-k+1}$, so that this sequence and the verification of $Y_{n+1} = 1$ are a good candidate having the first two conditions for linearly-faulty-and-logarithmically-provable scheme. A_i's uniform distribution can be replaced with the random oracle, and thus the above sequence satisfies the last "deterministic" condition heuristically by using a cryptographic hash function in the standard model.

Our Ephemeral-Nova Construction. Using the above idea for the recursive sequence and the verification procedure, we now construct the Ephemeral-Nova by slightly modifying the augmented function F' and the IVC procedure. As shown in Fig. 1 and Fig. 2, the execution of F' in each round consists of checking, folding, and invoking procedures. Although all these internal procedures of F' are necessary for proving knowledge-soundness, adding some redundant procedures may not affect knowledge-soundness. We extend F' to have two modes, which are determined by an additional input $\mathsf{b}_i \in \{0,1\}$. We call this extended F' a trapdoor augmented function F' and sometimes use the notation F'_{b_i} to denote $F'(\cdot, \mathsf{b}_i)$. According to bit b_i, the trapdoor F'_{b_i} sets

$$z_{i+1} := \begin{cases} F(z_i, \omega_i) \text{ if } \mathsf{b}_i = 1 \\ \quad z_i \qquad \text{if } \mathsf{b}_i = 0 \quad // \text{ {\scriptsize z_i can be replaced with any value except $F(z_i,\omega_i)$.}} \end{cases}$$

for some prime number q of the form $2^k\alpha + 1$ for some $k \geq \lambda$ and odd integer α. If $\mathsf{b}_i = 1$, this process is equivalent to the original F'. Otherwise, F' skips the execution of F. Therefore, we call the cases of $\mathsf{b}_i = 1$ and $\mathsf{b}_i = 0$ a normal mode and a trapdoor mode, respectively. The trapdoor F'_{b_i} additionally takes Y_i as input and F'_{b_i} updates Y_i according to the following rule.

$$Y_{i+1} := Y_i^{2\alpha} \cdot \big(\mathsf{b}_i + (1-\mathsf{b}_i)\cdot \mathsf{u}_i.\mathsf{x}\big) \pmod q \text{ and } Y_0 := 1.$$

Let $A_i = \big(\mathsf{b}_i + (1-\mathsf{b}_i)\cdot \mathsf{u}_i.\mathsf{x}\big)$. If $\mathsf{b}_i = 1$, then we have $A_i = 1$. Otherwise, A_i has a uniform distribution heuristically since $\mathsf{u}_i.\mathsf{x}$ is a hash value. From the analysis of the recursive sequence in Eq. (2), we know that $Y_{i+1} = 1$ could be a good verification procedure for linearly-faulty-logarithmically-provable IVC scheme. We provide a concrete description of the trapdoor augmented function F' and the Ephemeral-Nova in Fig. 4 and Fig. 5, respectively.

Choice of Prime Number q. The shape $\alpha \cdot 2^k + 1$ of the prime number q is essential in the construction of Ephemeral-Nova. By the prime number theorem, for fixed $k = O(\lambda)$, one can find $\alpha \cdot 2^k + 1$ prime by adjusting α in $O(\log \lambda)$ times.

3.1 Specific Attack to Ephemeral-Nova IVC

Now, we demonstrate a specific attack on the Ephemral-Nova IVC scheme in Fig. 5. For the sake of simplicity, we abuse the notation $F^{(t)}(z_i, \omega_i)$ to denote an output of t times F execution with k local inputs $\omega_i, \ldots \omega_{i+t-1}$ sequentially, i.e.

```
$F'_{\mathsf{b}_i} := F'(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, \boxed{Y_i, \mathsf{b}_i}) \to \mathsf{x}$:
    Compute $\boxed{z_{i+1}} := \begin{cases} F(z_i, \omega_i) & \text{if } \mathsf{b}_i = 1 \\ z_i & \text{if } \mathsf{b}_i = 0 \end{cases}$ // Any value except $F(z_i, \omega_i)$ can be used.
    If $i$ is 0, output $\mathsf{hash}(\mathsf{pp}, 1, z_0, F(z_0, \omega_i), \mathsf{u}_\perp, \boxed{Y_1})$; otherwise,
    1. check if $\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i, \boxed{Y_i})$, where $\mathsf{u}_i.\mathsf{x}$ is the public IO of $\mathsf{u}_i$,
    2. check if $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\perp.E, 1)$,
    3. compute $\mathsf{U}_{i+1} \leftarrow \mathsf{NIFS.V}(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, T)$, and
    4. output $\mathsf{hash}(\mathsf{pp}, i+1, z_0, z_{i+1}, \mathsf{U}_{i+1}, \boxed{Y_{i+1}})$, where
       $\boxed{Y_{i+1}} := Y_i^{2\alpha} \cdot \big(\mathsf{b}_i + (1 - \mathsf{b}_i) \cdot \mathsf{u}_i.\mathsf{x}\big) \pmod{q}$ and $Y_0 := 1$.
```

*We assume that appropriate *deterministic* encoding process is applied from the function value $\mathsf{hash}(\cdot)$ into $\mathbb{Z}_q$ for computing Y_{i+1}.

** The box □ indicates the difference from the original augmented function F'.

Fig. 4. Trapdoor Augmented Function F'

$F^{(t)}(z_i, \omega_i) = F(F(\cdots F(z_i, \omega_i), \omega_{i+1}), \cdots), \omega_{i+t-1})$. Suppose that $q = \alpha \cdot 2^k + 1$ is prime, the current time step is $n = k$, and collisions do not occur among F function values: $F^{(k)}(z_i, \omega_i) \neq z_j$ for all $i \neq j \leq k$.

The adversary creates a forgery $\widehat{\Pi_k}$ by running $\mathcal{P}_\mathsf{b}$ for each round in the following order $(\mathcal{P}_1, \mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_1)$ That is, the trapdoor mode F'_0 is used only in the $i = 1$ step and the normal mode F'_1 is used in all the other steps $i = [k] \backslash \{1\}$. Finally, after the last step, the IVC verifier $\mathcal{V}$ verifies $\widehat{\Pi_k}$ and the final output z_n.

It is clear that z_n is not equal to the correct value $F^{(k)}(z_0, \omega_0)$ since $\mathcal{P}_0$ skipped the execution of F and sets $z_2 = z_1$ so that $z_n = F^{(k-1)}(z_0, \omega_0)$. Nevertheless, we argue that the IVC verifier accepts the proof $\widehat{\Pi_k} = ((\widehat{\mathsf{U}_k}, \widehat{\mathsf{V}_k}), (\widehat{\mathsf{u}_k}, \widehat{\mathsf{v}_k}))$. In fact, both the trapdoor mode and the normal mode of F are correct executions of the augmented function F'. Therefore, both $(\widehat{\mathsf{U}_k}, \widehat{\mathsf{V}_k})$ and $(\widehat{\mathsf{u}_k}, \widehat{\mathsf{v}_k})$ are correct committed relaxed R1CS induced by F', where $(\widehat{\mathsf{U}_k}, \widehat{\mathsf{V}_k})$ are also corrected folded by folding scheme for F'. This allows $\widehat{\Pi_k}$ to pass the test in the third and fourth lines of the IVC verifier procedure in Fig. 5. Next, we check whether Y_k is equal to 1 or not. Since the current step is k, by Fermat's little theorem, we can confirm that $Y_k = (Y_2^{\alpha \cdot 2^k})^{\alpha^{k-1}} = 1$. Hence, the second line of the IVC verifier procedure will be passed.

3.2 Knowledge Soundness Proof in the log-bounded round IVC Model

We prove that Ephemeral-Nova has knowledge soundness in the log-bounded round IVC model.

```
$\underline{\mathcal{G}(1^\lambda) \to \mathsf{pp}}$: Output $\mathsf{pp} \leftarrow \mathsf{NIFS.G}(1^\lambda, N)$

$\underline{\mathcal{K}(\mathsf{pp}, F) \to (\mathsf{pk}, \mathsf{vk})}$:  1. Compute $\mathsf{pk}_{FS} \leftarrow \mathsf{NIFS.K}(\mathsf{pp}, \mathsf{s}_{F'})$
                        2. Output $(\mathsf{pk}, \mathsf{vk}) \leftarrow ((F, \mathsf{pk}_{FS}), (F, \mathsf{pp}))$

$\underline{\mathcal{P}_{\mathsf{b}}(\mathsf{pk}, (i, z_0, z_i), \omega_i, \Pi_i, \boxed{Y_i}) \to \Pi_{i+1}}$:
  $\underline{\text{Parse } \Pi_i \text{ as } ((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i)) \text{ and then}}$
    1. if $i$ is 0, set $Y_0 = 1$ and compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp.E)$;
       otherwise, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow \mathsf{NIFS.P}(\mathsf{pk}, (\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$
    2. compute $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1}) \leftarrow \mathsf{trace}(F', (\mathsf{vk}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, \boxed{Y_i, \mathsf{b}}))$, and
    3. output $\Pi_{i+1} \leftarrow ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.
$\underline{\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \Pi_i) \to \{0, 1\}}$:
  $\underline{\text{If } i \text{ is 0, check that } z_0 = z_i;}$
  otherwise,
    1. parse $\Pi_i$ as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$,
    2. check if $\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i, \boxed{1})$,
    3. check if $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\perp.E, 1)$, and
    4. check if $(\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, the committed relaxed R1CS induced by $F'$.
```

* The box □ indicates the difference from the original augmented function F'

** The instance-witness pair $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1})$ from the trace in $\mathcal{P}$ should follow the setting: $(\mathsf{u}_{i+1}.E, \mathsf{u}_{i+1}.s) = (\mathsf{u}_\perp.E, 1)$, .

Fig. 5. Ephemeral-Nova IVC

Theorem 2. *The IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}_1, \mathcal{V})$ in Fig. 5 satisfies perfect completeness and knowledge soundness if the non-interactive folding scheme* NIFS *satisfies perfect completeness and knowledge soundness.*

Due to space limitations, the full proof of Theorem 2 is included in Appendix A. Instead, we here sketch the proof idea. The Ephemeral-Nova is designed to be equivalent to Nova if the trigger is not activated. In particular, if we set $b = 1$, the augmented function F'_1, the IVC prover $\mathcal{P}_1$, and verifier $\mathcal{V}$ are essentially identical to the original Nova IVC, so that the Ephmeral-Nova IVC satisfies the completeness. For the knowledge soundness, it would be sufficient to show that passing the IVC verification guarantees that the trigger has not been activated. Because if the case, all the remaining proofs will be essentially equivalent to the original knowledge-soundness proof, by the design of the Ephemeral Nova.

Let us provide a brief idea about proving non-activation of trigger. We consider a log-round $n \leq \frac{\lambda}{2}$, where p is a λ-bit prime. We claim that if the IVC verifier $\mathcal{V}$ accepts the proof Π_n, then the skipping trigger cannot be activated during n-times computation $F^{(n)}$. When the trigger is activated (that is, $b = 0$) at i-th round, the additional indicator Y_i is changed to an arbitrary value because it is an output of hash. On the other hand, to give an acceptance from $\mathcal{V}$, the final additional input Y_n, which is an element in Π_n, should be equal to 1. By the construction of F'_{b_i} and uniform distribution of hash outputs, the additional value Y_i for all $i \in [n]$ should be equal to 1 without negligible probability. (Refer to lemma 4 in Appendix A.) This means that the trigger has not been activated during n times computation; therefore, we can rule out the case $b = 0$ and the remaining soundness proof is equivalent to that of the original Nova.

4 Model for Security Analysis

We first briefly review features of popular idealized models for group-based systems, and then set up an appropriate model for security analysis of the Nova IVC scheme.

Notation. We define notations for groups. Let $\mathbb{G}$ be an additive cyclic group of prime order p. When the group generator G is fixed, we use a bracket notation $[a]_G$ for a scalar $a \in \mathbb{Z}_p$ to denote the group element $a \cdot G$. If the generator is clear from the context, we often omit the subscript G and write as $[b] \in \mathbb{G}$. For $\boldsymbol{a} = (a_1, \ldots a_\ell) \in \mathbb{Z}_p^\ell$ and $[\boldsymbol{b}]_G = ([b_1]_G, \ldots, [b_\ell]_G) \in \mathbb{Z}_p^\ell$, a multi-scalar addition between $\boldsymbol{a}$ and $[\boldsymbol{b}]$ is denoted by $\langle \boldsymbol{a}, [\boldsymbol{b}] \rangle = \sum_{i=1}^{\ell} a_i \cdot [b_i]_G$.

Two Candidates: Generic Group Model and Algebraic Group Model The generic group model (GGM) is an idealized model where all group operations are carried out by making oracle queries [44, 48, 43, 42]. This model is designed to capture the behavior of natural general algorithms which operate independently of any particular group descriptions. In fact, this model is divided by a way to handle group elements. The adversary in Shoup's model [48] gets random-encoded values of the additive group $\mathbb{Z}_p$ which are considered as group elements, but the adversary in Maurer's model [43] cannot access to the value directly but get pointers indicating the line number in the oracle's table. Recently, Zhandry demonstrate the difference of these two models [50].

The algebraic group model (AGM), another idealized model proposed by Fuchsbauer, Kiltz, and Loss, requires whenever an algorithm outputs a group element G, it also outputs a representation $\boldsymbol{c}$ such that $\langle \boldsymbol{c}, \boldsymbol{G} \rangle = G$, where $\boldsymbol{G}$ is a vector of group elements the algorithm took as input [28]. In particular, a specific group description is fixed and known to all algorithms, and there is no oracle query for group operations in the AGM. The intuition of the AGM is to restrict algorithms to output a new group element G only by deriving it via group operations from known group elements. In fact, the concept of algebraic adversary has already been studied in several literature [10, 25, 46, 14, 29, 2, 1, 5, 37] and the AGM of Fuchsbauer, Kiltz, and Loss [28] is the first formal framework for security proofs with respect to algebraic adversaries.

The GGM and the AGM are two the most popular models for the analysis of group-based systems. We now present some limitations of two models, which are identified by either previous literature or ourselves, and slightly refine definitions for setting up an appropriate model for our purpose.

Limitation of GGM From the definition of GGM, it might cover a smaller class of algorithms than those in the AGM becasue algorithms are not allowed to use group descriptions. Another limitation of GGM, which is more critical to our purpose, is that the ideal group oracle cannot be instantiated to the arithmetic circuit. In the Nova IVC using group-based folding scheme, the folding process containing group operations is arithmetized and the arithmetized group operations are publicly accessible to all algorithms. That is, the adversary can access to the specific group description from this arithmetization. In fact, the same issue occurs when we use the arithmetized cryptographic hash function which is modeled as the random oracle. Then, the resulting security analysis should rely on the heuristic AGM instantiation in the standard model. We avoid heuristic analysis as much as possible, so that we move to the next candiate, the AGM.

Limitation of AGM The AGM is proposed as a model lying between the standard model and the GGM, and it is one of main reasons why the AGM has received so much attention recently [33, 3, 35, 8, 32].

Recently Zhang, Zhou, and Katz found a counter example to this claim about positioning of the AGM by presenting a computational problem that is hard under the discrete logarithm (DL) assumption in the AGM but easily solvable in the GGM [51]. This implies that the class of algorithms considered in AGM does not cover all generic algorithms. Furthermore, Zhandry proved the uninstantiability of AGM by presenting an one-time MAC that is secure in the AGM under the DL assumption, but insecure in the standard model [50]. That is, the AGM might not cover some potential threat in the standard model. Finally, the AGM does not completely capture a situation often occurred in the context of IVC such that a group element is encoded by multiple encodings.

We pointed out three limitations above. Now, we consider each one by one to refine the AGM to adapt to the security analysis of the Nova IVC.

First AGM Refinement We first consider the counter example of Zhang, Zhou, and Katz [51]. Let σ is an encoding of $\mathbb{Z}_p$ for group element and $\mathcal{A}$ be an adversarial algorithm. The binary encoding game is defined as follow.

```
1. choose $z \xleftarrow{\$} \mathbb{Z}_p$ and parse $Z = \sigma(z)$ as the bitstring $z_1 \cdots z_\ell$.
2. $(G, H_1, \ldots, H_\ell) := (\sigma(1), \sigma(z_1), \ldots, \sigma(z_\ell))$
3. $Z' \leftarrow \mathcal{A}(G, H_1, \ldots, H_\ell)$
4. Return 1 iff $(Z' = Z)$.
```

In [51], it was proven infeasible for algebraic adversary $\mathcal{A}$ to win under the DL assumption. However, a *non-algebraic-but-generic* attack exists by simply setting $z'_i := 1$ iff $G = H_i$ and then outputting $Z' := z'_1 \cdots z'_\ell$. That is, the input distribution $(G, H_1, \ldots, H_\ell)$ informs all bits of Z' and the adversary just gathers these bits and returns by concatenating them. All processes of $\mathcal{A}$ are generic operation, but finally $\mathcal{A}$ outputs a group element Z' without knowing its discrete log with base G. All algebraic adversary fails to win this game because it is necessary for algebraic adversary to output the representation of Z' together.

The main reason of studying algebraic adversaries is that most known cryptographic algorithms fall in this category; in most cryptographic applications, non-generic generation of a new group element is usually not helpful for solving (attacking, resp.) hard problems (crpytosystems, resp.). However, the above binary encoding game is contrivedly created that the solution Z is indeed embedded into the distribution of the input. In fact, a similar problem is already pointed out by Fuchsbauer, Kiltz, and Loss [28]; the adversary takes a bitstring of $X' = X \| \perp$ as the input and is required to output X. It is trivial for generic adversary to win the game, but we can prove its hardness against algebraic adversaries under the discrete logarithm (DL) assumption because any algebraic algorithm should output the discrete log of X as well, but it cannot under the DL assumption. The example of Fuchsbauer, Kiltz, and Loss is essentially the same as the bit encoding game in the sense that the solution X (Z, resp.) is embedded into the input to the adversary X' (the distribution of group elements, resp.). To rule out this example, Fuchsbauer, Kiltz, and Loss restricts that for the inputs to algebraic adversaries, they syntactically distinguish group elements from other inputs and require that the latter does not depend on any group elements. However, the bit encoding game bypasses this requirements by using only group elements in the input to the adversary, but successfully embeds information about Z into the inputs.

The requirement of Fuchsbauer, Kiltz, and Loss was insufficient to prevent side information embedded into the inputs. Nevertheless, if we restrict our attention to the *common random string model*, where all group elements are uniformly and independently generated, then the side information cannot be embedded into the distribution of group element as in the binary encoding game. To analyze the security of the Nova IVC using group-based folding scheme, we use only Pedersen commitment scheme and thus all the commitment keys are the common random strings. Therefore, non-algebraic-but-generic attacks similar to that identified in [51] do not exist in the Nova IVC security game and the analysis in the AGM is still meaningful.

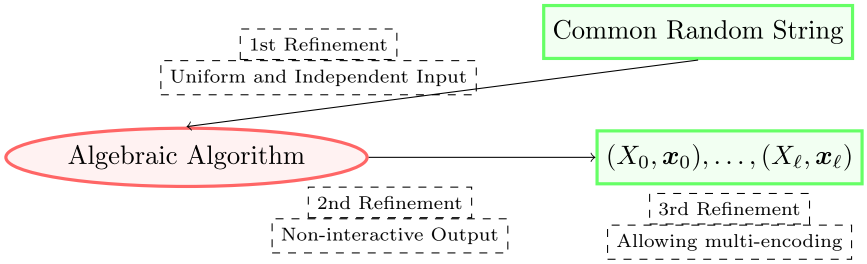
Second AGM Refinement We discuss about the AGM un-instantiability of Zhandry [50]. Zhandry proposed a one-time MAC in the AGM such that an efficient attack exists in the standard model, but no algebraic adversary can break the scheme under the discrete logarithm assumption. The idea of the standard-model adversary is to get a bitstring of a new group element X by using one-time MAC query. Then, the adversary does not know the discrete log of X. If the adversary is algebraic adversary, it should find the discrete log of X because X is not considered as a group element when the adversary gets. Therefore, under the DL assumption, all algebraic adversaries fail to break the system.

As in the bit encoding game, a non-algebraic generation of group element (by the one-time MAC query) is essential to design non-algebraic adversary in Zhandry's construction. However, at least in *non-interactive* protocol, the adversary cannot obtain additional side information except public parameters, so that we can rule out similar attacks to Zhandry's standard-model attacks.

Third AGM Refinement First, we consider a situation we can easily face with in the context of the IVC but not thoroughly considered in the original AGM. Let $\mathcal{A}$ be an algebraic algorithm for a group $\mathbb{G}$ such that $\mathcal{A}$ takes $\boldsymbol{G} \in \mathbb{G}^\ell$ as input and outputs a pair of group element and its representation $(X_0, \boldsymbol{x}_0) \in \mathbb{G} \times \mathbb{Z}_p^\ell$ such that $X_0 = \langle \boldsymbol{x}_0, \boldsymbol{G} \rangle$. Although $\boldsymbol{x}_0$ is not a group element but a vector of field elements in $\mathbb{Z}_p^\ell$, we may apply a pre-declared bijection to identify a part of $\boldsymbol{x}_0$ a group element. For example, consider $\boldsymbol{x}_0$ as a bitstring, extract pre-determined positions of bits, and check the resulting bitstring is a group element or not; For example, for elliptic curve group, the GMT checks whether a given pair $(x, y) \in \mathbb{Z}_q \times \mathbb{Z}_q$ satisfies an elliptic curve equation $Y^2 = X^3 + aX + b$ over $\mathbb{Z}_q$. For each element in $\mathbb{Z}_q$, one can use a $\lceil \log q \rceil$-bitstring with zero-padding for the most significant bits. If the way to identify group elements is pre-declared in the system parameter settings, and the algebraic algorithm already knows this, how should the algebraic algorithm act? This situation is equivalent to allow for algorithms to use two distinct encodings for each group element. In fact, a random bitstring has only negligible chance to be considered as a group element if $\log p > \lambda$ because for each x, there exist at most two ys satisfying the elliptic curve equation. Using two encodings does not increase the probability that the algorithm non-algebraically generates a new group element in accident. Therefore, it is reasonable to ask the algebraic algorithm to return the corresponding representation of the group element embedded in $\boldsymbol{x}_0$. To clarify algebraic algorithm's behaviour in this situation, we add some clarification as follows.

1. Algorithms are allowed to use multiple (constant number of) encodings for group elements if those are pre-declared in the pubic parameters.
2. Group elements in the output of algorithms are syntactically distinguish from other elements; that is, given a bitstring in the ouput, the bit-positions and the encoding method applied for group element are defined in the system parameter.

Thus, if the represent vector $\boldsymbol{x}_0$ contains an encoding of another group element X_1, the algebraic adversary should output the corresponding representation vector $\boldsymbol{x}_1$ as well. It can be repeated until the i-th representation vector $\boldsymbol{x}_i$ does not contain a group element.



In summary, to avoid situations where creating new group elements without knowing the discrete logarithms can help the adversary win the game, we limit our interest to the following situations: (1st Refinement) All group elements in the input to the algorithm are common random strings (2nd Refinement) Adversarial algorithms are non-interactive, so queries to obtain additional information are not allowed. In addition, in order to cover a broad class of algorithms, we extend our interest to the following situation: (3rd refinement) Pre-declared multiple encodings are allowed. In fact, the first and second refinements are too restrictive since, for example, there exist interactive algebraic algorithms with structured reference strings that could have benefit from the analysis in the AGM. Nevertheless, these refinements are determined basically to cover the circumstance of the Nova IVC. It could be possible to find a milder condition than these constraints that can cover a wider range of situations without contradiction to the results in [51, 50]. We leave it as an open question.

5 Zero-Testing Hash Functions

The group-based folding scheme in [40] is knowledge-sound under the DL assumption and it can be made non-interactive in the random oracle model using the Fiat-Shamir transformation [27]. However, to use non-interactive folding scheme in the Nova IVC, the folding verifier should be arithmetized and thus the random oracle should be instantiated in the standard model, by using a concrete hash function. There are studies [20, 19] that aim at removing the heuristic instantiations of random oracles by introducing new variants of random oracles. We propose a different approach for avoiding heuristic analysis because we do not want to change the Nova IVC construction but rather provide a new soundness analysis. To this end, we propose a new plausible property of cryptographic hash functions such as SHA-256 that is sufficient for proving the knowledge soundness in the AGM. Note that the new property of hash function we introduce is an intractability property, like preimgage-resistance and collision-resistance. That is, this property of hash function cannot solely replace the random oracles because it cannot replace such a power of the random oracle to extract witness by rewinding algorithms. However, it can be combined with the AGM to fully replace the random oracles in the proof of the Nova IVC.

5.1 Zero-Testing Property of Hash Functions

In the context of proof systems, a polynomial is often used to prove several relations at the same time. For example, to prove three equalities $A_i = B_i$ for $i = 0, 1, 2$, one can prove a polynomial $\mathsf{poly} = \sum_i (A_i - B_i) X^i$ is identical to zero. In interactive protocols, the Schwartz-Zippel lemma enables to statistically verify it; (1) Prover commmits to the polynomial, (2) a random challenge r is chosen by the verifier, (3) check $\mathsf{poly}(r) \overset{?}{=} 0$. In non-interactive protocols the Fiat-Shamir transformation applied, the second step can be changed with hash evaluation and check if $\mathsf{poly}(\mathsf{hash}(\mathsf{poly})) \overset{?}{=} 0$, where hash is considered as the random oracle. In the random oracle model, we can rewind the prover multiple times with a fixed commitment. Therefore, poly passing the test implies that poly vanishes at multiple points larger than the degree of poly, so that it is identical to zero. Although this argument in the non-interactive protocol is well analyzed in the random oracle model, we believe that, even without the random oracle model, it is still reasonable to expect that cryptographic hash function also guarantees this method of testing zero polynomial. We formalize this belief in Definition 6. Let λ be the security parameter and hash be a cryptographic hash function that maps to $\mathbb{Z}_p$, where p is a prime of length $O(\lambda)$.

Definition 6. *(Zero-Testing) For a hash function* hash*, we say that* hash *has the* **zero-testing property** *if it is infeasible for any PPT adversary to find a non-zero polynomial* $\mathsf{poly} \in \mathbb{Z}_p[X]$ *of degree* $O(\lambda)$ *that satisfies* $\mathsf{poly}(\mathsf{hash}(\mathsf{poly})) = 0 \pmod p$ *except a negligible probability.*

In fact, the above zero-testing property is too simple to directly apply to various cryptosystems. We provide this to help the readers understand the intuition behind the following generalization of the zero testing property.

Definition 7. *(General Zero-Testing) Let* C *be a binding commitment and* D *be an arbitrary deterministic function that maps from a domain* $\mathcal{D}$ *to* $\mathbb{Z}_p[X]$ *of degree* $O(\lambda)$*. For a hash function* hash*, we say that* hash *has the* **general zero-testing property** *if no PPT adversary can find* $d \in \mathcal{D}$ *and auxiliary input* τ*, with non-negligible probability, such that* $D(d)$ *is a non-zero polynomial and* $\mathsf{D}(d)(\mathsf{hash}(\mathsf{C}(d), \tau)) = 0 \pmod p$*.*

Note that the general zero-testing property is equivalent to the zero-testing property if we set $\mathcal{D} = \mathbb{Z}_p[X]$, both C and D to be identity maps, and $\tau = \emptyset$. To support the reliability of the (general) zero-testing property, we prove that at least the random oracles satisfy the (general) zero-testing property.

Lemma 1. *The random oracle* hash *has the zero-testing property.*

Proof. For each hash query poly, the hash result $\mathsf{hash}(\mathsf{poly})$ is uniformly random, so that the probability $\mathsf{poly}(\mathsf{hash}(\mathsf{poly})) = 0 \pmod p$ holds is at most $\deg(\mathsf{poly})/p$. For q distinct queries, all query results are mutually independent and thus the probability that at least one equality holds is bounded by the sum probability $\frac{q \deg(\mathsf{poly})}{p}$, which is still negligible in λ. □

Note that the above proof does not rely on the programmability of the random oracle, but uses only the uniform and independent distribution of the random oracle outputs.

Lemma 2. *If* C *is Pedersen commitments with binding property, then the random oracle* hash *has the general zero-testing property in the AGM.*

Proof. The basic proof strategy is identical to Lemma 1, except that we additionally require the ability that for each query (c, τ), we can see d such that the adversary used to compute $\mathsf{C}(d) = c$. If we have such an ability, then for each hash query (c, τ), we can specify d and thus the polynomial $D(d)$ the adversary used. The remaining analysis is the same in the proof of Lemma 1.

Now we argue that we have such an ability against the algebraic adversary. For each query c, c consists group elements, so that the algebraic adversary should output the corresponding representation based on the commitment key of the Pedersen commitment. Because of the binding property, such the representation is exactly openings d of Pedersen commitment scheme such that $\mathsf{C}(d) = c$, and thus $D(d)$ is the polynomial the adversary used. □

5.2 Schnorr's NIZK in the AGM

As a warm-up example to show effectiveness of the zero-testing property, we present a new knowledge-soundness proof of Schnorr's NIZK protocol, which is one of the simplest proof knowledge protocol; it proves that (G, H) is an instance of the relation $\mathcal{R} = \{(G, [x]_G; x \in \mathbb{Z}_p)\}$.

Prover

1. chooses $k \xleftarrow{\$} \mathbb{Z}_p$ and computes $K := [k]_G$.
2. computes $e \leftarrow \mathsf{hash}(G, H, K)$.
3. computes $s = k + ex \bmod p$ and outputs (s, K).

Verifier accepts if,

given (s, K), $[s]_G \overset{?}{=} K + e \cdot H$ holds, where $e \leftarrow \mathsf{hash}(G, H, K)$.

Using the general zero-testing property, we can prove that Shnorr's non-interactive protocol is knowledge-sound in the AGM. In particular, the extraction is tight and the random oracles are not required.

Theorem 3. *If* hash *has the general zero-testing property, then the Schnorr's non-interactive protocol satisfies the knowledge soundness in the AGM. In particular, the running time of extractor is equivalent to the running time of the algebraic prover, except constant operations.*

Proof. Given an arbitrary algebraic prover $\mathcal{P}^*$, we construct an extractor $\mathcal{E}$ that extracts the witness x. $\mathcal{P}^*$ begins with taking a pair of (G, H) as input. Suppose that $\mathcal{P}^*$ outputs a proof (s, K) that passes the verification; that is, the equality $s \cdot G = K + e \cdot H$ holds where $e \leftarrow \mathsf{hash}(G, H, K)$. Since $\mathcal{P}^*$ is an algebraic adversary, it should output the representation (k_1, k_2) of the group element K such that $K = k_1 \cdot G + k_2 \cdot H$. Thus, we have $(s - k_1) \cdot G = (k_2 + e) \cdot H$, so that we get the discrete logarithm of H as $x = (s - k_1) \cdot (k_2 + e)^{-1} \pmod p$ unless $k_2 + e = 0 \pmod p$.

Now we argue that $k_2 \neq -e \pmod p$. Suppose that $k_2 = -e \pmod p$. Then, $\mathsf{hash}(G, H, k_1 \cdot G - e \cdot H)$ is a solution of a polynomial $e - X = 0 \pmod p$. Using the notations d, C, and D in the general zero-testing property, we can set $d = (k_1, -e)$, (G, H) is the commitment key of C, and $\mathsf{D}(k_1, -e) = e - X \in \mathbb{Z}_p[X]$, where D discards k_1. Therefore, no PPT algorithm can find $d = (k_1, -e)$

that satisfies $\mathsf{D}(d)(\mathsf{hash}(G, H, \mathsf{C}(d))) = 0 \pmod p$ by the general zero-testing property, so that $k_2 \neq -e \pmod p$.

What the extractor did except running $\mathcal{P}^*$ is only to compute constant operations $x = (s - k_1) \cdot (k_2 + e)^{-1} \pmod p$. □

6 New Soundness Analysis of Nova IVC with Group-based Folding Scheme

Pedersen Commitment for Vectors. Pedersen commitment scheme is a homomorphic commitment scheme with perfect hiding and computational binding properties under the discrete logarithm assumption. The setup algorithm $\mathsf{Setup}(1^\lambda, \ell)$ takes the dimension variable ℓ and outputs the commitment key ck consisting of an $(\ell + 1)$-dimensional vector $\mathbb{G}^{\ell+1}$. The message $\boldsymbol{x}$ is an ℓ-dimensional vector in $\mathbb{Z}_p^\ell$. The commitment to $\boldsymbol{x}$ with a random scalar $r \xleftarrow{\$} \mathbb{Z}_p$ is computed as a multi-scalar addition $\langle \boldsymbol{x} \| r, \mathsf{ck} \rangle \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r)$. The homomorphic property is naturally induced from the characteristic of the cyclic group $\mathbb{G}$.

Group-based Folding Scheme from [40]. In [40], the group-based non-interactive folding scheme $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ for the committed relaxed R1CS relation $\mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}}$ in Eq. (1) is proposed, where the public parameter pp_{FS} is generated by G and the common structure s is taken as an input of K. The folding prover $\mathsf{NIFS.P}$ takes two committed relaxed R1CS instance-witness pairs, and outputs a folded instance-witness pair (u, v), with prover's transcript T. The folding verifier $\mathsf{NIFS.V}$ takes two committed relaxed R1CS instances u_1, u_2, and T, and then outputs a folded instance u. We provide a full description of such group-based folding scheme in Fig. 6.

Looking at the Knowledge Soundness Proof of Nova [40]. In this paragraph, we briefly review the knowledge soundness proof of Nova [40]. The premise of the proof is the knowledge soundness of the internal non-interactive folding scheme in the standard model that assumes the existence of the extractor $\tilde{\mathcal{E}}$ satisfying condition in Def. 3. To construct the IVC extractor $\mathcal{E}$, which outputs $(\omega_0, \ldots, \omega_{n-1})$, the proof follows a general recursive proof strategy. That is, $\mathcal{E}$ inductively generates $\mathcal{E}_i$ that, given $\mathcal{E}_{i+1}$, outputs $(z_i, \ldots, z_{n-1})$, $(\omega_i, \ldots, \omega_{n-1})$ and Π_n. In fact, $\mathcal{E}_{i+1}$ directly implies an adversarial folding prover $\tilde{\mathcal{A}}_i$ for the i-th round and $\mathcal{E}_i$ can be constructed from $\tilde{\mathcal{A}}_i$. In the procedure of $\mathcal{E}_i$, the folding extractor $\tilde{\mathcal{E}}_i$ of $\tilde{\mathcal{A}}_i$ is additionally called, so that we have inequality between running times of algorithms as follows.

$$\mathtt{time}(\mathcal{E}_i) > \mathtt{time}(\tilde{\mathcal{E}}_i) + \mathtt{time}(\tilde{\mathcal{A}}_i) > 2 \cdot \mathtt{time}(\mathcal{E}_{i+1})$$

if $\mathtt{time}(\tilde{\mathcal{E}}_i) > \mathtt{time}(\tilde{\mathcal{A}}_i)$. Then, $\mathtt{time}(\mathcal{E})$ increases exponentially in n. The soundness proof of Nova paper relies on the assumption about the knowledge soundness of non-interactive folding scheme in the standard model when the random

$\underline{\mathsf{G}(1^\lambda, N=(m,\ell)) \to \mathsf{pp}_{FS}}$: Output commitment keys $\mathsf{pp}_{\boldsymbol{e}} \overset{\$}{\leftarrow} \mathbb{G}^m$ and $\mathsf{pp}_{\boldsymbol{w}} \overset{\$}{\leftarrow} \mathbb{G}^{m-\ell-1}$.

$\underline{\mathsf{K}(\mathsf{pp}_{FS}, \mathsf{s}=(A,B,C)) \to \mathsf{pk}_{FS}}$: output $\mathsf{pk}_{FS} \leftarrow (\mathsf{pp}_{FS}, \mathsf{s})$.

$\underline{\mathsf{NIFS.P}(\mathsf{pk}_{FS}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \to (\mathsf{u}, \mathsf{v}), T}$:

1. For $i = 1, 2$, parse $(\mathsf{u}_i, \mathsf{v}_i) = ((E_i, s_i, W_i, \mathsf{x}_i), (\boldsymbol{e}_i, r_{\boldsymbol{e}_i}, \boldsymbol{w}_i, r_{\boldsymbol{w}_i}))$ and then set $\boldsymbol{z}_i = (\boldsymbol{w}_i, \mathsf{x}_i, s_i)$
2. Compute $\boldsymbol{t} = A\boldsymbol{z}_1 \circ B\boldsymbol{z}_2 + A\boldsymbol{z}_2 \circ B\boldsymbol{z}_1 - s_1 \cdot C\boldsymbol{z}_2 - s_2 \cdot C\boldsymbol{z}_1$.
3. Pick $r_{\boldsymbol{t}} \overset{\$}{\leftarrow} \mathbb{Z}_p$ and compute $T = \langle (\boldsymbol{t}, r_{\boldsymbol{t}}), \mathsf{pp}_{\boldsymbol{e}} \rangle$.
4. $r \leftarrow \mathsf{hash}(\mathsf{u}_1, \mathsf{u}_2, T)$.
5. Output $T, \mathsf{u} := (E, s, W, \mathsf{x})$ and $\mathsf{v} := (\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}})$ where

$$E \leftarrow E_1 + r \cdot T + r^2 \cdot E_2 \qquad W \leftarrow W_1 + r \cdot W_2 \tag{4}$$
$$s \leftarrow s_1 + r \cdot s_2 \qquad \mathsf{x} \leftarrow \mathsf{x}_1 + r \cdot \mathsf{x}_2 \tag{5}$$
$$\boldsymbol{e} \leftarrow \boldsymbol{e}_1 + r \cdot \boldsymbol{t} + r^2 \cdot \boldsymbol{e}_2 \qquad \boldsymbol{w} \leftarrow \boldsymbol{w}_1 + r \cdot \boldsymbol{w}_2 \tag{6}$$
$$r_{\boldsymbol{e}} \leftarrow r_{\boldsymbol{e}_1} + r \cdot r_{\boldsymbol{t}} + r^2 \cdot r_{\boldsymbol{e}_2} \qquad r_{\boldsymbol{w}} \leftarrow r_{\boldsymbol{w}_1} + r \cdot r_{\boldsymbol{w}_2} \tag{7}$$

$\underline{\mathsf{NIFS.V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, T) \to \mathsf{u}}$:

1. For $i = 1, 2$, parse $\mathsf{u}_i = (E_i, s_i, W_i, \mathsf{x}_i)$
2. $r \leftarrow \mathsf{hash}(\mathsf{u}_1, \mathsf{u}_2, T)$
3. Output $\mathsf{u} := (E, s, W, \mathsf{x})$ satisfying Eq. (4) and Eq. (5).

Fig. 6. Group-based Non-Interactive Folding Scheme in [38]

oracle is instantiated with a cryptographic hash function. Considering the corresponding interactive folding scheme (or non-interactive scheme in the random oracle model), $\tilde{\mathcal{E}}_i$ uses the rewinding strategy with the forking lemma so that $\texttt{time}(\tilde{\mathcal{E}}_i) > \texttt{time}(\tilde{\mathcal{A}}_i)$ holds. To avoid blowing up, we do not reduce to the knowledge soundness of the folding scheme, so that the $\texttt{time}(\mathcal{E}_i)$ only additionally increases without using the folding extractor.

6.1 Polynomially-Long Rounds Soundness of Nova IVC in the AGM

In this section, we prove the knowledge soundness of the Nova IVC scheme (Fig. 3) in the AGM. Following the group-based folding scheme, instance-witness pairs $(\mathsf{U}_i, \mathsf{V}_i)$ and $(\mathsf{u}_i, \mathsf{v}_i)$ are represented as follows:

$$\mathsf{U}_i = (E_i^{(1)}, s_i^{(1)}, W_i^{(1)}, \mathsf{x}_i^{(1)}),\ \mathsf{u}_i = (E_i^{(2)}, s_i^{(2)}, W_i^{(2)}, \mathsf{x}_i^{(2)}) \in \mathbb{G} \times \mathbb{Z}_p \times \mathbb{G} \times \mathbb{Z}_p$$
$$\mathsf{V}_i = (\boldsymbol{e}_i^{(1)}, r_{\boldsymbol{e}_i}^{(1)}, \boldsymbol{w}_i^{(1)}, r_{\boldsymbol{w}_i}^{(1)}),\ \ \mathsf{v}_i = (\boldsymbol{e}_i^{(2)}, r_{\boldsymbol{e}_i}^{(2)}, \boldsymbol{w}_i^{(2)}, r_{\boldsymbol{w}_i}^{(2)}) \in \mathbb{Z}_p^m \times \mathbb{Z}_p \times \mathbb{Z}_p^{m-\ell-1} \times \mathbb{Z}_p,$$

Pre-declared encoding for committed relaxed R1CS of augmented execution F'. Basically, we use an ordinary encoding for group elements. In addition, we allow an additional encoding for group elements, defined as follows: Let us consider the structure of the committed relaxed R1CS relation for augmented execution F' in Fig. 2. F'-circuit includes the computation part for $z_{i+1} = F(z_i, \omega_i)$ and the folding part for $\mathsf{U}_{i+1} \leftarrow \mathsf{NIFS.V}(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, T)$. Therefore, the witness v_{i+1} of the $(i+1)$-th committed relaxed R1CS contains the internal values of F' such as i-th inputs z_i, ω_i and commitments $(E_i^{(1)}, W_i^{(1)}, E_i^{(2)}, W_i^{(2)})$. By definition, the witness v_{i+1} forms a representation vector in $\mathbb{Z}_p$. On the other hand, v_{i+1} contains group elements such as $(E_i^{(1)}, W_i^{(1)}, E_i^{(2)}, W_i^{(2)})$ as v_{i+1} is the witness of the F'-R1CS; that is, each group element is encoded into a $\mathbb{Z}_p$-vector, which is our second encoding for group elements. Based on the premise of our AGM model, if the algebraic algorithm outputs a proof $\Pi_{i+1} = ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$ that passes the IVC verification, then v_{i+1} contains group elements $E_i^{(1)}, W_i^{(1)}, E_i^{(2)}$, $W_i^{(2)}$ and we assume that those group elements are efficiently identified by using the pre-declared second encoding method.

Set of Representation Vectors. The adversary $\mathcal{P}^*$ outputs a forgery proof Π_n. If $\mathcal{P}^*$ is an algebraic algorithm, it should output the corresponding representations to group elements in Π_n. If those representations again contain another group element, $\mathcal{P}^*$ should output its representation as well. Let us use a notation R to denote a set of all these representations associated with a sequentially embedded group elements into Π_n.

We present our new statement for knowledge soundness of the Nova IVC and its proof below.

Theorem 4. *If* hash *has the general zero-testing property and the discrete logarithm assumption holds in* $\mathbb{G}$*, then the Nova IVC scheme in Fig. 3 combined with the group-based folding scheme based on* $\mathbb{G}$ *and* hash *(Fig. 6) satisfies the definition 2 in the AGM.*

Proof. Let $\mathcal{P}^*$ be an expected polynomial time algebraic adversary against knowledge soundness for the arbitrary polynomial step $n = poly(\lambda)$. Then, $\mathcal{P}^*$ takes public parameters $(\mathsf{pp}_{\boldsymbol{e}}, \mathsf{pp}_{\boldsymbol{w}})$, which consist of randomly chosen group elements, and outputs the initial value z_0, n-th output value z, and IVC proof Π with set R of representation vectors following multi-encoding. Using the adversary $\mathcal{P}^*$, we construct the expected polynomial time extractor $\mathcal{E}$, which takes (z_0, z) and outputs a sequence of local inputs $(\omega_i)_{i \in [n-1]}$ satisfying $z_n = z$ and $F(z_{i-1}, \omega_{i-1}) = z_i$ for all $i \in [n]$ from $\mathcal{P}^*$.

We first construct i-th partial extractor $\mathcal{E}_i$ for all $i \in [n]$. Concretely, $\mathcal{E}_i$ takes sequences $(z_i, \ldots, z_n)$ and $(\omega_i, \ldots, \omega_{n-1})$, and accepting i-th IVC proof $\Pi_i = ((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ with additional representations. Then, $\mathcal{E}_i$ outputs ω_{i-1} and Π_{i-1}. After constructing $(\mathcal{E}_i)$, we construct the total extractor $\mathcal{E}$ using $(\mathcal{E}_i)$ following descending order. We describe the partial extractor algorithm $\mathcal{E}_i$ and total extractor algorithm $\mathcal{E}$ in Fig. 7.

To claim that the partial extractor $\mathcal{E}_i$ can extract the previous proof Π_{i-1}, we give the following lemma.

```
$\underline{\mathcal{E}_i(\mathsf{pp}, (z_i, \dots, z_n), (\omega_i, \dots, \omega_{n-1}), \mathit{\Pi}_i, \mathsf{R}) \to z_{i-1}, \omega_{i-1}, \mathit{\Pi}_{i-1}}$:
    1. Parse $\mathit{\Pi}_i = ((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$
    2. Retrieve $z_{i-1}, \omega_{i-1}, \mathsf{U}_{i-1}, \mathsf{u}_{i-1}$ from $\mathsf{v}_i$
    3. Retrieve $E^{(1)}_{i-1}, W^{(1)}_{i-1}, E^{(2)}_{i-1}, W^{(2)}_{i-1}$ from $\mathsf{U}_{i-1}$ and $\mathsf{u}_{i-1}$
    4. Retrieve representation vectors $(\tilde{\boldsymbol{e}}^{(1)}_{i-1}, \tilde{\boldsymbol{w}}^{(1)}_{i-1}), (\tilde{\boldsymbol{e}}^{(2)}_{i-1}, \tilde{\boldsymbol{w}}^{(2)}_{i-1}) \in \mathbb{Z}_p^{m+1} \times \mathbb{Z}_p^{m-\ell}$ of
       $(E^{(1)}_{i-1}, W^{(1)}_{i-1}), (E^{(2)}_{i-1}, W^{(2)}_{i-1})$ from representation set $\mathsf{R}$
    5. Set $\mathsf{V}_{i-1} = (\tilde{\boldsymbol{e}}^{(1)}_{i-1}, \tilde{\boldsymbol{w}}^{(1)}_{i-1})$ and $\mathsf{v}_{i-1} = (\tilde{\boldsymbol{e}}^{(2)}_{i-1}, \tilde{\boldsymbol{w}}^{(2)}_{i-1})$
    6. Output $z_{i-1}, \omega_{i-1}, \mathit{\Pi}_{i-1} = ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$

$\underline{\mathcal{E}(\mathsf{pp}, \rho) \to (\omega_0, \dots, \omega_{n-1})}$:
    1. $z_0, z_n, \mathit{\Pi}_n, \mathsf{R} \leftarrow \mathcal{P}^*(\mathsf{pp}, \rho)$
    2. Parse $\mathit{\Pi}_n = ((\mathsf{U}_n, \mathsf{V}_n), (\mathsf{u}_n, \mathsf{v}_n))$
    3. Retrieve $(z_{n-1}, \omega_{n-1})$ from $\mathsf{v}_n$
    4. For $i = n - 1$ to $1$ :
           $z_{i-1}, \omega_{i-1}, \mathit{\Pi}_{i-1} \leftarrow \mathcal{E}_i(\mathsf{pp}, (z_i, \dots, z_n), (\omega_i, \dots, \omega_{n-1}), \mathit{\Pi}_i, \mathsf{R})$
    5. Output $(\omega_0, \dots, \omega_{n-1})$
```

Fig. 7. The description of the IVC Extractor $\mathcal{E}$

Lemma 3. *Assume that $(z_i, \dots, z_n)$ and $(\omega_i, \dots, \omega_{n-1})$ are two sequences satisfying $z_{j+1} = F(z_j, w_j)$ for all $j = i \dots n-1$, R is a set of representation vectors outputted by $\mathcal{P}^*$, and $\mathit{\Pi}_i = ((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ be a valid IVC proof (Fig. 3) combined with the group-based folding scheme (Fig. 6). If the hash in group-based folding scheme has general zero test property and the discrete logarithm assumption holds in the underlying group $\mathbb{G}$, then there is a PPT $\mathcal{E}_i$ that outputs a valid IVC proof $\mathit{\Pi}_{i-1} = ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$ and $(i-1)$-th inputs z_{i-1} and ω_{i-1} satisfying $z_i = F(z_{i-1}, w_{i-1})$.*

Proof. Following the group-based folding scheme, instance-witness pairs $(\mathsf{U}_i, \mathsf{V}_i)$ and $(\mathsf{u}_i, \mathsf{v}_i)$ form the following description:

$$\mathsf{U}_i = (E^{(1)}_i, s^{(1)}_i, W^{(1)}_i, \mathsf{x}^{(1)}_i),\ \mathsf{u}_i = (E^{(2)}_i, s^{(2)}_i, W^{(2)}_i, \mathsf{x}^{(2)}_i) \in \mathbb{G} \times \mathbb{Z}_p \times \mathbb{G} \times \mathbb{Z}_p$$
$$\mathsf{V}_i = (\boldsymbol{e}^{(1)}_i, r^{(1)}_{\boldsymbol{e}_i}, \boldsymbol{w}^{(1)}_i, r^{(1)}_{\boldsymbol{w}_i}),\ \ \mathsf{v}_i = (\boldsymbol{e}^{(2)}_i, r^{(2)}_{\boldsymbol{e}_i}, \boldsymbol{w}^{(2)}_i, r^{(2)}_{\boldsymbol{w}_i}) \in \mathbb{Z}_p^m \times \mathbb{Z}_p \times \mathbb{Z}_p^{m-\ell-1} \times \mathbb{Z}_p,$$

By our premise, each $(\mathsf{U}_i, \mathsf{V}_i)$ and $(\mathsf{u}_i, \mathsf{v}_i)$ belongs to a committed relaxed R1CS relation, so we can represent the instance-witness pairs as follows:

$$E^{(1)}_i = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{e}}, \boldsymbol{e}^{(1)}_i; r^{(1)}_{\boldsymbol{e}_i}) = \langle \boldsymbol{e}^{(1)}_i \parallel r^{(1)}_{\boldsymbol{e}_i}, \mathsf{pp}_{\boldsymbol{e}} \rangle, \tag{8}$$
$$W^{(1)}_i = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{w}}, \boldsymbol{w}^{(1)}_i; r^{(1)}_{\boldsymbol{w}_i}) = \langle \boldsymbol{w}^{(1)}_i \parallel r^{(1)}_{\boldsymbol{w}_i}, \mathsf{pp}_{\boldsymbol{w}} \rangle, \tag{9}$$
$$E^{(2)}_i = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{e}}, \boldsymbol{e}^{(2)}_i; r^{(2)}_{\boldsymbol{e}_i}) = \langle \boldsymbol{e}^{(2)}_i \parallel r^{(2)}_{\boldsymbol{e}_i}, \mathsf{pp}_{\boldsymbol{e}} \rangle, \tag{10}$$
$$W^{(2)}_i = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{w}}, \boldsymbol{w}^{(2)}_i; r^{(2)}_{\boldsymbol{w}_i}) = \langle \boldsymbol{w}^{(2)}_i \parallel r^{(2)}_{\boldsymbol{w}_i}, \mathsf{pp}_{\boldsymbol{w}} \rangle. \tag{11}$$

$\mathcal{E}_i$'s goal is to compute $\mathit{\Pi}_{i-1} = ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$ and (z_{i-1}, ω_{i-1}) satisfying $z_i = F(z_{i-1}, w_{i-1})$. In fact, if $\mathcal{E}_i$ successfully extracts a valid proof

$\mathit{\Pi}_{i-1}$, $(\mathsf{u}_{i-1}, \mathsf{v}_{i-1})$ should belong to the original R1CS relation for F'. That is, the $(i-1)$-th witness v_{i-1} contains input and output of the $(i-1)$-th F-execution $z_i = F(z_{i-1}, \omega_{i-1})$. Therefore, $\mathcal{E}_i$ can efficiently extract (z_{i-1}, ω_{i-1}) from $\mathit{\Pi}_{i-1}$, and thus, we will now focus on how to construct a valid proof $\mathit{\Pi}_{i-1}$ from $\mathit{\Pi}_i$.

We first consider two instances U_{i-1} and u_{i-1}. Since $\mathit{\Pi}_i$ is a valid proof, v_i is the i-th witness of the original R1CS induced by F', and thus contains the advice $(\mathsf{pp}, \mathsf{U}_{i-1}, \mathsf{u}_{i-1}, (i, z_0, z_{i-1}), \omega_{i-1}, T_{i-1})$ of F'. Therefore $\mathcal{E}_i$ can efficiently extract U_{i-1} and u_{i-1} from v_i.

Next, we show how $\mathcal{E}_i$ extracts witness V_{i-1} and v_{i-1} such that they together with $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ can pass the IVC verification in Fig. 3. To satisfy the fourth line of the IVC verification, at least V_{i-1} and v_{i-1} should be openings of the corresponding instances U_{i-1} and u_{i-1}, respectively. Using the notation defined above, V_{i-1} and v_{i-1} should be the openings of $E_{i-1}^{(j)}$ and $W_{i-1}^{(j)}$, respectively.

Because $E_{i-1}^{(j)}$ and $W_{i-1}^{(j)}$ are group elements, their representations should exist in the set R. Then, $\mathcal{E}_i$ can efficiently find the corresponding representation vectors $\tilde{\boldsymbol{e}}_{i-1}^{(1)}$, $\tilde{\boldsymbol{w}}_{i-1}^{(1)}$, $\tilde{\boldsymbol{e}}_{i-1}^{(2)}$, and $\tilde{\boldsymbol{w}}_{i-1}^{(2)}$

of $E_{i-1}^{(1)}, W_{i-1}^{(1)}, E_{i-1}^{(2)}$ and $W_{i-1}^{(2)}$ under base $(\mathsf{pp}_{\boldsymbol{e}}, \mathsf{pp}_{\boldsymbol{w}})$. Let the extractor $\mathcal{E}_i$ set $\mathsf{V}_i = (\tilde{\boldsymbol{e}}_{i-1}^{(1)}, \tilde{\boldsymbol{w}}_{i-1}^{(1)})$ and $\mathsf{v}_i = (\tilde{\boldsymbol{e}}_{i-1}^{(2)}, \tilde{\boldsymbol{w}}_{i-1}^{(2)})$

Now, we claim that the IVC verifier $\mathcal{V}$ accepts the extracted proof $\mathit{\Pi}_{i-1}$ from $\mathcal{E}_i$. To this end, we first check the fourth step of the IVC verifier such that $(\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, and then show that it passes the second and third steps of the IVC verifier. The CR-R1CS relation $\mathcal{R}_{\mathsf{pp},\mathsf{s}}$ consists of the opening-checks, which are the first two equations of $\mathcal{R}_{\mathsf{pp},\mathsf{s}}$,and the R1CS-like relation, which is the last two equations of $\mathcal{R}_{\mathsf{pp},\mathsf{s}}$. To show $(\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, we first show those that satisfy the opening-checks and then show that the witness satisfies the R1CS-like relation.

We know that, by the hypothesis, $(\mathsf{u}_i, \mathsf{v}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$ is an original R1CS instance-witness pair that involves the folding verifier $\mathsf{NIFS.V}$ whose input and output are $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1}, T_{i-1})$ and U_i respectively. Therefore, we know that the following relation holds.

$$\begin{array}{ll} E_i^{(1)} = E_{i-1}^{(1)} + rT_{i-1} + r^2 E_{i-1}^{(2)}, & s_i^{(1)} = s_{i-1}^{(1)} + rs_{i-1}^{(2)}, \\ W_i^{(1)} = W_{i-1}^{(1)} + rW_{i-1}^{(2)}, & \mathsf{x}_i^{(1)} = \mathsf{x}_{i-1}^{(1)} + r\mathsf{x}_{i-1}^{(2)} \end{array} \tag{12}$$

where r is hash outputs in $\mathsf{NIFS.V}$. In fact, T_{i-1} is also a group element contained in the witness v_i, so we can find the corresponding representation, say $\tilde{\boldsymbol{t}}_{i-1}$, from R. In addition, considering the extracted witness $(\tilde{\boldsymbol{e}}_{i-1}^{(1)}, \tilde{\boldsymbol{w}}_{i-1}^{(1)}), (\tilde{\boldsymbol{e}}_{i-1}^{(2)}, \tilde{\boldsymbol{w}}_{i-1}^{(2)})$, we know that those are representation vectors with base $\mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}}$. That is, the following equation holds.

$$\begin{array}{l} E_{i-1}^{(1)} = \langle \tilde{\boldsymbol{e}}_{i-1}^{(1)}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle,\ W_{i-1}^{(1)} = \langle \tilde{\boldsymbol{w}}_{i-1}^{(1)}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle, \\ E_{i-1}^{(2)} = \langle \tilde{\boldsymbol{e}}_{i-1}^{(2)}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle,\ W_{i-1}^{(2)} = \langle \tilde{\boldsymbol{w}}_{i-1}^{(2)}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle \\ T_{i-1} = \langle \tilde{\boldsymbol{t}}_{i-1}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle \end{array} \tag{13}$$

Combining Eq. (12), Eq. (13) with Eq. (8) and Eq. (9), we obtain the following linear relations.

$$\begin{aligned}\langle \boldsymbol{e}_i^{(1)} \parallel r_{\boldsymbol{e}_i}^{(1)}, \mathsf{pp}_{\boldsymbol{e}} \rangle &\overset{(8)}{=} E_i^{(1)} \overset{(12)\&(13)}{=} \langle \tilde{\boldsymbol{e}}_{i-1}^{(1)} + r\tilde{\boldsymbol{t}}_{i-1} + r^2 \tilde{\boldsymbol{e}}_{i-1}^{(2)}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle \\ &\overset{\text{DL assump.}}{=} \langle \tilde{\boldsymbol{e}}_{i-1}^{(1)} + r\tilde{\boldsymbol{t}}_{i-1} + r^2 \tilde{\boldsymbol{e}}_{i-1}^{(2)}, \mathsf{pp}_{\boldsymbol{e}} \rangle, \\ \langle \boldsymbol{w}_i^{(1)} \parallel r_{\boldsymbol{w}_i}^{(1)}, \mathsf{pp}_{\boldsymbol{w}} \rangle &\overset{(9)}{=} W_i^{(1)} \overset{(12)\&(13)}{=} \langle \tilde{\boldsymbol{w}}_{i-1}^{(1)} + r\tilde{\boldsymbol{w}}_{i-1}^{(2)}, \mathsf{pp}_{\boldsymbol{e}} \| \mathsf{pp}_{\boldsymbol{w}} \rangle \\ &\overset{\text{DL assump.}}{=} \langle \tilde{\boldsymbol{w}}_{i-1}^{(1)} + r\tilde{\boldsymbol{w}}_{i-1}^{(2)}, \mathsf{pp}_{\boldsymbol{w}} \rangle,\end{aligned}$$

Let the representation vectors parse to two parts as follows:

$$\begin{aligned}\tilde{\boldsymbol{e}}_{i-1}^{(1)} &= \bar{\boldsymbol{e}}_{i-1}^{(1)} \| \bar{r}_{\boldsymbol{e}_{i-1}}^{(1)}, \quad \tilde{\boldsymbol{e}}_{i-1}^{(2)} = \bar{\boldsymbol{e}}_{i-1}^{(2)} \| \bar{r}_{\boldsymbol{e}_{i-1}}^{(2)} \in \mathbb{Z}_p^m \times \mathbb{Z}_p, \\ \tilde{\boldsymbol{w}}_{i-1}^{(1)} &= \bar{\boldsymbol{w}}_{i-1}^{(1)} \| \bar{r}_{\boldsymbol{w}_{i-1}}^{(1)}, \tilde{\boldsymbol{w}}_{i-1}^{(2)} = \bar{\boldsymbol{w}}_{i-1}^{(2)} \| \bar{r}_{\boldsymbol{w}_{i-1}}^{(2)} \in \mathbb{Z}_p^{m-\ell-1} \times \mathbb{Z}_p\end{aligned} \tag{14}$$

By the Eq. (13) and Eq. (14), we finally get the following desired relation for the opening-checks of committed relaxed R1CS:

$$\begin{aligned}E_{i-1}^{(1)} &= \mathsf{Com}(\mathsf{pp}_{\boldsymbol{e}}, \bar{\boldsymbol{e}}_{i-1}^{(1)}; \bar{r}_{\boldsymbol{e}_{i-1}}^{(1)}), \; W_{i-1}^{(1)} = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{w}}, \bar{\boldsymbol{w}}_{i-1}^{(1)}; \bar{r}_{\boldsymbol{w}_{i-1}}^{(1)}), \\ E_{i-1}^{(2)} &= \mathsf{Com}(\mathsf{pp}_{\boldsymbol{e}}, \bar{\boldsymbol{e}}_{i-1}^{(2)}; \bar{r}_{\boldsymbol{e}_{i-1}}^{(2)}), \; W_{i-1}^{(2)} = \mathsf{Com}(\mathsf{pp}_{\boldsymbol{w}}, \bar{\boldsymbol{w}}_{i-1}^{(2)}; \bar{r}_{\boldsymbol{w}_{i-1}}^{(2)}).\end{aligned}$$

To complete the claim $(\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, we showed the opening-checks and the R1CS-like relation is remained. From the hypothesis $(\mathsf{U}_i, \mathsf{V}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$ and Eq. (12), we can derive the following equality.

$$\begin{aligned}0 &= A\boldsymbol{z} \circ B\boldsymbol{z} - sC\boldsymbol{z} - \boldsymbol{e} \\ &= A(\boldsymbol{z}_1 + r\boldsymbol{z}_2) \circ B(\boldsymbol{z}_1 + r\boldsymbol{z}_2) - (s_1 + rs_2)C(\boldsymbol{z}_1 + r\boldsymbol{z}_2) - (\boldsymbol{e}_1 + r\boldsymbol{t} + r^2\boldsymbol{e}_2) \\ &= A\boldsymbol{z}_1 \circ B\boldsymbol{z}_1 - s_1C\boldsymbol{z}_1 - \boldsymbol{e}_1 + r^2(A\boldsymbol{z}_2 \circ B\boldsymbol{z}_2 - s_2C\boldsymbol{z}_2 - \boldsymbol{e}_2) + r\delta(\boldsymbol{z}_1, \boldsymbol{z}_2, A, B)\end{aligned}$$

where $\boldsymbol{t} = \tilde{\boldsymbol{t}}_{i-1}, s = s_i^{(1)}, s_b = s_{i-1}^{(b)}$, $\boldsymbol{z} = (\boldsymbol{w}_i^{(1)}, \mathsf{x}_i^{(1)}, s_i^{(1)})$, $\boldsymbol{e} = \boldsymbol{e}_i^{(1)}$, $\boldsymbol{z}_b = (\boldsymbol{w}_{i-1}^{(b)}, \mathsf{x}_{i-1}^{(b)}, s_{i-1}^{(b)})$, $\boldsymbol{e}_b = \boldsymbol{w}_{i-1}^{(b)}$ for $b \in \{1, 2\}$ and $\delta(\boldsymbol{z}_1, \boldsymbol{z}_2, A, B)$ is a redundant term consisting $\boldsymbol{z}_1$, $\boldsymbol{z}_2$, A, and B. We argue that the general zero test property of hash guarantees that each coefficient of r^j-term should be zero without negligible probability; The last term of the above equation can be considered as a degree-2 polynomial in r whose coefficients are determined by $d := (\boldsymbol{z}_1, \boldsymbol{e}_1, \boldsymbol{z}_2, \boldsymbol{e}_2, \boldsymbol{t})$ with A, B, C. We also know that r is the hash value of $\mathsf{u}_{i-1}, \mathsf{U}_{i-1}$ and T_{i-1}, which can be considered as commitments to d with binding property.

Therefore, we finally obtain the following equation:

$$A\boldsymbol{z}_1 \circ B\boldsymbol{z}_1 - s_1C\boldsymbol{z}_1 - \boldsymbol{e}_1 = 0 = A\boldsymbol{z}_2 \circ B\boldsymbol{z}_2 - s_2C\boldsymbol{z}_2 - \boldsymbol{e}_2$$

and we can conclude $(\mathsf{u}_{i-1}, \mathsf{v}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$.

We now consider the second and third lines of the IVC verification $\mathcal{V}$ in Fig. 3. By the hypothesis, $\mathit{\Pi}_i$ is a valid IVC proof. It implies that $(\mathsf{u}_i, \mathsf{v}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$ is an instance-witness pair of the original R1CS for F'-execution. By $\mathcal{P}$ algorithm in Fig. 3, we know that $(\mathsf{u}_i, \mathsf{v}_i)$ is constructed from $\mathsf{trace}(F', (\mathsf{vk}, \mathsf{U}_{i-1}, \mathsf{u}_{i-1}, (i, z_0, z_{i-1}),$

$\omega_{i-1}, T)$). Because the invocation of F' in Fig. 2 contains hash and error commitment/scalar check, we know that the following equation holds, which is exactly what we wanted to show.

$$\begin{aligned} \mathsf{u}_{i-1}.\mathsf{x} &= \mathsf{hash}(\mathsf{pp}, i-1, z_0, z_{i-1}, \mathsf{U}_{i-1}) \\ (\mathsf{u}_{i-1}.E, \mathsf{u}_{i-1}.s) &= (\mathsf{u}_{\perp}.E, 1) \end{aligned}$$

Finally, we conclude that $\mathcal{V}(\mathsf{vk}, (i, z_0, z_{i-1}), \Pi_{i-1}) = 1$. □

By the lemma 3, $\mathcal{E}_i$ outputs accepting $(i-1)$-th proof Π_{i-1}. From running of $\mathcal{E}_i$, $\mathcal{E}$ can extract the sequence of $(\omega_0, \ldots, \omega_{n-1})$ with $z_{i+1} = F(z_i, \omega_i)$ for all $i = 0, \ldots n-1$. Because the running time of each $(\mathcal{E}_i)_{i \in [n-1]}$ and adversary $\mathcal{P}^*$ are polynomial time, the total running time of $\mathcal{E}$ is bound polynomial time, where $n = poly(\lambda)$. □

7 Concluding Remarks

In this paper, we showed that an unnecessary redundant procedure in the augmented function F' may be used as a trigger for attacks that are activated only at a predetermined time. To investigate this type of attacks to the Nova IVC scheme, it is necessary to prove the knowledge soundness for polynomial rounds. We presented the first provable security analysis of Nova IVC's knowledge soundness for polynomial rounds. In particular, our proof does not rely on the heuristic random oracle instantiation but newly introduced hash function with general zero-testing property. There are interesting open questions. There are many other IVC schemes that also have soundness proof only for log rounds. It would be interesting to study polynomial rounds security of those schemes. In particular, our AGM refinement may be helpful if the schemes are group-based. Next, it would be also interesting to find another security proof for Nova IVC in the standard model.

References

1. Michel Abdalla, Fabrice Benhamouda, and Philip MacKenzie. Security of the J-PAKE password-authenticated key exchange protocol. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 571–587. IEEE Computer Society, 2015.
2. Masayuki Abe, Jens Groth, and Miyako Ohkubo. Separating short structure-preserving signatures from non-interactive assumptions. In *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 628–646. Springer, 2011.
3. Balthazar Bauer, Georg Fuchsbauer, and Julian Loss. A classification of computational assumptions in the algebraic group model. In *CRYPTO (2) 2020*, volume 12171 of *LNCS*, pages 121–151. Springer, 2020.
4. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. *Algorithmica*, 79:1102–1160, 2017.
5. David Bernhard, Marc Fischlin, and Bogdan Warinschi. On the hardness of proving cca-security of signed elgamal. In *PKC (1) 2016*, volume 9614 of *LNCS*, pages 47–69. Springer, 2016.

6. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *ITCS12*, LNCS, pages 326–349. Springer, 2012.
7. Dan Boneh, Benedikt Bünz, and Ben Fisch. A survey of two verifiable delay functions. *Cryptology ePrint Archive*, 2018.
8. Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Efficient polynomial commitment schemes for multiple points and polynomials. *Cryptology ePrint Archive, Report 2020/81*, page 81, 2020.
9. Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Proof-carrying data from additive polynomial commitments. In *CRYPTO (1) 2021*, volume 12825 of *LNCS*, pages 649–680. Springer, 2021.
10. Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In *EUROCRYPT 1998*, volume 1403 of *LNCS*, pages 59–71. Springer, 1998.
11. Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Coda: Decentralized cryptocurrency at scale. *Cryptology ePrint Archive*, 2020.
12. Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Mina: Decentralized cryptocurrency at scale. *New York Univ. O (1) Labs, New York, NY, USA, Whitepaper*, pages 1–47, 2020.
13. Sean Bowe, Jack Grigg, and Daira Hopwood. Recursive proof composition without a trusted setup. *Cryptology ePrint Archive*, 2019.
14. Emmanuel Bresson, Jean Monnerat, and Damien Vergnaud. Separation results on the "one-more" computational problems. In *CT-RSA 2008,*, volume 4964 of *LNCS*, pages 71–87. Springer, 2008.
15. Benedikt Bünz and Binyi Chen. Protostar: Generic efficient accumulation/folding for special sound protocols. *Cryptology ePrint Archive*, 2023.
16. Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In *CRYPTO (1) 2021*, volume 12825 of *LNCS*, pages 681–710. Springer, 2021.
17. Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data from accumulation schemes. *Cryptology ePrint Archive*, 2020.
18. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent SNARKs from DARK compilers. In *EUROCRYPT 2020*, volume 12105 of *LNCS*, pages 677–706. Springer, 2020.
19. Megan Chen, Alessandro Chiesa, Tom Gur, Jack O'Connor, and Nicholas Spooner. Proof-carrying data from arithmetized random oracles. In *EUROCRYPT (2) 2023*, volume 14005, pages 379–404, 2022.
20. Megan Chen, Alessandro Chiesa, and Nicholas Spooner. On succinct non-interactive arguments in relativized worlds. In *EUROCRYPT (2) 2022*, volume 13276, pages 336–366, 2022.
21. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zksnarks with universal and updatable srs. In *EUROCRYPT 2020*, volume 12105 of *LNCS*, pages 738–768. Springer, 2020.
22. Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In *EUROCRYPT 2020 Part I*, volume 12105 of *LNCS*, pages 769–793. Springer, 2020.
23. Alessandro Chiesa and Eran Tromer. Proof-carrying data and hearsay arguments from signature cards. In *ICS*, volume 10, pages 310–331, 2010.
24. Coda. https://codaprotocol.com.

25. Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 272–287. Springer, 2002.
26. Ethereum (ETH) Blockchain Explorer. https://etherscan.io.
27. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO 1986*, volume 263 of *LNCS*, pages 186–194. Springer, 1987.
28. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *CRYPTO (2) 2018*, volume 10992 of *LNCS*, pages 33–62. Springer, 2018.
29. Sanjam Garg, Raghav Bhaskar, and Satyanarayana V. Lokam. Improved bounds on security reductions for discrete log based signatures. In *CRYPTO 2008*, volume 5157 of *LNCS*, pages 93–107. Springer, 2008.
30. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct nizks without pcps. In *EUROCRYPT 2013*, pages 626–645. Springer, 2013.
31. Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 99–108, 2011.
32. Ashrujit Ghoshal and Stefano Tessaro. Tight state-restoration soundness in the algebraic group model. In *CRYPTO (3) 2021*, volume 12827 of *LNCS*, pages 64–93. Springer, 2021.
33. Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In *CCS '20*, pages 2007–2023. ACM, 2020.
34. Assimakis Kattis and Joseph Bonneau. Proof of necessary work: Succinct state verification with fairness guarantees. *Cryptology ePrint Archive*, 2020.
35. Jonathan Katz, Julian Loss, and Jiayu Xu. On the security of time-lock puzzles and timed commitments. In *TCC (3) 2020*, volume 12552 of *LNCS*, pages 390–413. Springer, 2020.
36. Dmitry Khovratovich, Mary Maller, and Pratyush Ranjan Tiwari. Minroot: Candidate sequential function for ethereum vdf. *Cryptology ePrint Archive*, 2022.
37. Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In *CRYPTO (2) 2016*, volume 9815 of *LNCS*, pages 33–61. Springer, 2016.
38. Abhiram Kothapalli and Srinath Setty. Supernova: Proving universal machine executions without universal circuits. *Cryptology ePrint Archive*, 2022.
39. Abhiram Kothapalli and Srinath Setty. Hypernova: Recursive arguments for customizable constraint systems. *Cryptology ePrint Archive*, 2023.
40. Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In *CRYPTO (4) 2022*, volume 13510 of *LNCS*, pages 359–388. Springer, 2022.
41. Jonathan Lee, Kirill Nikitin, and Srinath Setty. Replicated state machines without replicated execution. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 119–134. IEEE, 2020.
42. Ueli Maurer, Christopher Portmann, and Jiamin Zhu. Unifying generic group models. *Cryptology ePrint Archive, Report 2020/996*, page 996, 2020.
43. Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In *10th IMA International Conference on Cryptography and Coding*, volume 3796, pages 1–12. Springer, 2005.

44. V. I. Nechaev. Complexiy of a determinate algorithm for the discrete logarithm. In *Mathematical Notes*, volume 55(2), pages 165–172, 1994.
45. Wilson Nguyen, Dan Boneh, and Srinath Setty. Revisiting the nova proof system on a cycle of curves. *Cryptology ePrint Archive*, 2023.
46. Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 1–20. Springer, 2005.
47. Srinath Setty. Spartan: Efficient and general-purpose zksnarks without trusted setup. In *CRYPTO 2020*, LNCS. Springer, 2020.
48. Victor Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT 1997*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997.
49. Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *TCC 2008*, volume 4948 of *LNCS*, pages 1–18. Springer, 2008.
50. Mark Zhandry. To label, or not to label (in generic groups). In *CRYPTO 2022*, volume 13509, pages 66–96, 2022.
51. Cong Zhang, Hong-Sheng Zhou, and Jonathan Katz. An analysis of the algebraic group model. In *ASIACRYPT 2022*, volume 13794, pages 310–322, 2022.

A Knowledge Soundness Proof of Ephemeral-Nova in the log-bounded round IVC Model

Proof. (*Completeness*): We argue that the $\mathcal{P}$'s output $\mathit{\Pi}_{i+1}$ from the execution F with $(i+1, z_0, z_{i+1}, \mathit{\Pi}_i)$ is valid proof if the i-th proof $\mathit{\Pi}_i$ is valid. Let $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$, F, and $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F')$ be the public parameters, an IVC execution, and prover/verifier key, respectively. Now, we claim that the IVC proof, which satisfies $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \mathit{\Pi}_i) = 1$ and $\mathcal{P}(\mathsf{pk}, i, z_0, z_i, \omega_i, \mathit{\Pi}_i) \to \mathit{\Pi}_{i+1}$, implies $\mathcal{V}(\mathsf{vk}, i, z_0, z_{i+1}, \mathit{\Pi}_{i+1}) = 1$, where $z_{i+1} = F(z_i, \omega_i)$. We consider two cases in which the step index i is equal to 0 or not.

Case $(i = 0)$: According to our premise, we know that $\mathit{\Pi}_0$ is a trivial valid proof $((\mathsf{u}_\perp, \mathsf{v}_\perp), (\mathsf{u}_\perp, \mathsf{v}_\perp))$. Now, we consider the validity of the updated proof $\mathit{\Pi}_1$. Let $\mathcal{P}_1$ take the input $(\mathsf{pk}, (1, z_0, z_1, \omega_0, \mathit{\Pi}_0, Y_0)$ and then get $\mathit{\Pi}_1$. From the $\mathcal{P}_1$ in Fig. 3, we obtain

$$\mathit{\Pi}_1 = ((\mathsf{u}_\perp, \mathsf{v}_\perp), (\mathsf{u}_1, \mathsf{v}_1))$$

where $(\mathsf{u}_1, \mathsf{v}_1)$ is R1CS instance-witness pair for F'_1 execution. Following the execution F'_1 in 4, we know

$$\mathsf{u}_1.\mathsf{x} = \mathsf{hash}(\mathsf{vk}, 1, z_0, F(z_0, \omega_0), \mathsf{u}_\perp, Y_1) \text{ where } Y_1 = Y_0^2 = 1 \qquad (15)$$

$$(\mathsf{u}_1.E, \mathsf{u}_1.s) = (\mathsf{u}_\perp.E, 1) \qquad (16)$$

From Eq. (15) and Eq. (16), second and third verifier conditions in Fig. 5 hold. To check the fourth condition, we only consider $(\mathsf{u}_1, \mathsf{v}_1) \in \mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}_{\mathsf{F}'}}$ because $(\mathsf{U}_1, \mathsf{V}_1) = (\mathsf{u}_\perp, \mathsf{v}_\perp)$ is already belong in the relation. From the tracing of F', $(\mathsf{u}_1, \mathsf{v}_1)$ should belong to the committed relaxed R1CS relation. Therefore, we can conclude that the IVC verifier accepts the following proof $\mathit{\Pi}_1$, $\mathcal{V}(\mathsf{pp}, 1, z_0, z, \mathit{\Pi}_1) = 1$.

<u>**Case** $(i \geq 1)$</u>: Suppose that Π_i is a valid IVC proof for verification $\mathcal{V}$ and Π_{i+1} be a proof generated by $\mathcal{P}_1$ with input $(\mathsf{pk}, (i, z_0, z_i, \omega_i, \Pi_i, Y_i)$

Based on the completeness of the underlying folding scheme and the premise that $(\mathsf{u}_i, \mathsf{v}_i)$ and $(\mathsf{U}_i, \mathsf{V}_i)$ are satisfying instance-witness pairs of the relation, we have $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1})$ is a satisfying instance-witness pair of the relation, i.e. $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}) \in \mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}_{\mathsf{F}'}}$.

From the tracing of F' execution with input $(\mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, Y_i, 1)$, we have that $\mathsf{u}_{i+1}.\mathsf{x} = \mathsf{hash}(\mathsf{pp}, i+1, z_0, z_{i+1}, \mathsf{U}_{i+1}, Y_{i+1})$ where $Y_{i+1} = Y_i^2 = 1$ and $(\mathsf{u}_{i+1}.E, \mathsf{u}_{i+1}.s) = (\mathsf{u}_{\perp}.E, 1)$. Therefore, the verifier $\mathcal{V}$ should accept the IVC proof $\Pi_{i+1} = ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

(*Knowledge Soundness*): For fixed step n, let the security parameter λ satisfy the following inequality: $\frac{\lambda}{2} \geq n$ and p be a λ-bit prime number. First, we claim that if the IVC verifier accepts the proof Π_n of n times execution F', then all execution types of i-th step should be F'_1 with high probability.

Let $j-1$ be the latest step of execution F' with the choice bit $\mathsf{b} = 0$. In this case, $Y_j = Y_{j-1}^{2\alpha} \cdot \mathsf{u}_i.\mathsf{x}$ can be viewed as a uniform random sample from $\mathbb{Z}_p^*$ because $\mathsf{u}_i.\mathsf{x}$ is an image of hash. From our hypothesis regarding the latest step, Y_n can be described by the following equation:

$$Y_n = Y_j^{(2\alpha)^{n-j}} \tag{17}$$

Due to the premise of acceptance by $\mathcal{V}$ in Fig. 5, the following relation holds: $\mathsf{u}_n.\mathsf{x} = \mathsf{hash}(\mathsf{pp}, n, z_0, z_n, \mathsf{U}_n, 1)$. On the other hand, the R1CS relationship for F' constrains that the last input of hash is Y_n. Therefore, $Y_n = 1$ holds with high probability. To claim that the probability of $Y_j^{(2\alpha)^{n-j}} = 1$ is $\mathsf{negl}(\lambda)$, let us consider the following Lemma 4.

Lemma 4. *Let $p = \alpha \cdot 2^\lambda$ be a prime with odd integer α. If integer n satisfies $\frac{\lambda}{2} \geq n$, then the following probability equation holds.*

$$\Pr_{x \xleftarrow{\$} \mathbb{Z}_p^*}[x^{(2\alpha)^n} = 1] \leq 2^{-\frac{\lambda}{2}} \tag{18}$$

Proof. Since the multiplicative group $\mathbb{Z}_p^*$ has order $\alpha \cdot 2^k$, the α^n-power subgroup $H := \{x^{\alpha^n} | x \in \mathbb{Z}_p^*\}$ has 2^k distinct elements. From the subgroup H, we can describe the probability as:

$$\Pr_{x \xleftarrow{\$} \mathbb{Z}_p^*}[x^{(2\alpha)^n} = 1] = \Pr_{x \xleftarrow{\$} \mathbb{Z}_p^*}[(x^{\alpha^n})^{2^n} = 1] = \Pr_{y \xleftarrow{\$} H}[y^{2^n} = 1]$$

To get upper bound of the probability $\Pr_{y \xleftarrow{\$} H}[y^{2^n} = 1]$, let us consider the upper bound of total number of $y \in H$ such that $y^{2^n} = 1$. If $y \in H$ satisfies $y^{2^n} = 1$, y should be a root of the polynomial $X^{2^n} - 1 \in \mathbb{Z}_p[X]$. By the fundamental theorem of algebra, $X^{2^n} - 1 \in \mathbb{Z}_p[X]$ has at most 2^n distinct roots, which means that the number of ys is at most 2^n. Therefore, the probability $\Pr_{y \xleftarrow{\$} H}[y^{2^n} = 1]$ is at most $\frac{2^n}{2^\lambda} = \frac{1}{2^{\lambda - n}} \leq 2^{-\frac{\lambda}{2}}$ □

By Lemma 4 and our premise $\frac{\lambda}{2} \geq n$, we can conclude that the probability of $Y_j^{(2\alpha)^{n-j}} = 1$ is negligible. For this reason, the probability of the case $\mathsf{b} = 0$ for any i-step is $\mathsf{negl}(\lambda)$. Then, we can consider that all execution types of i-th step should be F_1' with the exception of negligible probability.

Now, we only consider that augmented execution is F_1'. The following process is similar to soundness proof of Nova-IVC [40].

Let $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$. Consider an expected polynomial-time adversary $\mathcal{P}^*$ that outputs a function F on input pp, and let $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F)$. Suppose that, for a constant $n \leq \lambda$, $\mathcal{P}^*$ outputs (z_0, z, Π) such that

$$\mathcal{V}(\mathsf{vk}, n, z_0, z, \Pi) = 1.$$

We must construct an expected polynomial-time extractor $\mathcal{E}$ that with input (pp, z_0, z), outputs $(\omega_0, \ldots, \omega_{n-1})$ such that by computing for all $i \leq n$

$$z_i \leftarrow F(z_{i-1}, \omega_{i-1})$$

and $z_n = z$ with the exception of the probability $\mathsf{negl}(\lambda)$.

We show inductively that $\mathcal{E}$ can run an expected polynomial-time extractor $\mathcal{E}_i(\mathsf{pp})$ that outputs $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i)$ such that for all $j \in \{i + 1, \ldots n\}$,

$$z_j = F(z_{j-1}, \omega_{j-1})$$

and

$$\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1 \tag{19}$$

for $z_n = z$ with the exception of the probability $\mathsf{negl}(\lambda)$.

$\mathcal{E}$ run $\mathcal{E}_n$ first, and then using $\mathcal{E}_n$, construct $\mathcal{E}_{n-1}$ and repeat this process until reaching $\mathcal{E}_0$.

First, $\mathcal{E}_n(\mathsf{pp}, \rho)$ outputs $(\perp, \perp, \Pi_n)$, where Π_n is the output of $\mathcal{P}^*(\mathsf{pp}, \rho)$. Assume that $\mathcal{E}_n$ succeeds to get valid proof Π_n from IVC adversary $\mathcal{P}^*$.

For $i \geq 1$, suppose $\mathcal{E}$ can construct an expected polynomial-time extractor $\mathcal{E}_i$ that outputs $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}))$, and Π_i that satisfies the inductive hypothesis. To construct an extractor $\mathcal{E}_{i-1}$, $\mathcal{E}$ first constructs an adversary $\mathcal{A}_{i-1}$ for the non-interactive folding scheme as follows:
$\underline{\tilde{\mathcal{A}}_{i-1}(\mathsf{pp}, \rho)}$:

1. Let $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i) \leftarrow \mathcal{E}_i(\mathsf{pp}, \rho)$.
2. Parse Π_i as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$.
3. Parse v_i to retrieve $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1}, T_{i-1})$.
4. Output $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ and $((\mathsf{U}_i, \mathsf{V}_i), T_{i-1})$.

By the inductive hypothesis, we have that $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1$, where $\Pi_i \leftarrow \mathcal{E}_i(\mathsf{pp})$ with the exception of negligible probability $\mathsf{negl}(\lambda)$. Therefore, by the verifier's checks we have that $(\mathsf{u}_i, \mathsf{v}_i)$ and $(\mathsf{U}_i, \mathsf{V}_i)$ are satisfying instance-witness pairs, and that

$$\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i, Y_i)$$

Because $\mathcal{V}$ ensures that $(\mathsf{u}_i.E, \mathsf{u}_i.u) = (\mathsf{u}_\perp.E), 1)$, we have that v_i is indeed a satisfying assignment for F'. Then, by the construction of F' and the binding property of the hash function, we have that

$$\mathsf{U}_i = \mathsf{NIFS.V}(\mathsf{vk}, \mathsf{U}_{i-1}, \mathsf{u}_{i-1}, T_{i-1})$$

with the exception of negligible probability $\mathsf{negl}(\lambda)$. Thus, $\mathcal{A}$ succeeds in producing an accepting folded instance-witness pair $(\mathsf{U}_i, \mathsf{V}_i)$, for instances $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$, with the exception of $\mathsf{negl}(\lambda)$. Thus, $\mathcal{A}$ succeeds in producing an accepting folded instance-witness pair $(\mathsf{U}_i, \mathsf{V}_i)$, for instances $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ in expected polynomial-time.

Given an expected polynomial-time $\tilde{\mathcal{A}}_{i-1}$ and an expected polynomial-time folding scheme extractor $\tilde{\mathcal{E}}_{i-1}, \mathcal{E}$ constructs an expected polynomial time $\mathcal{E}_{i-1}$ as follows
$\underline{\mathcal{E}_{i-1}(\mathsf{pp}, \rho)}$:

```
1. $((\mathsf{U}_{i-1}, \mathsf{u}_{i-1}), (\mathsf{U}_i, \mathsf{V}_i), T_{i-1}) \leftarrow \tilde{\mathcal{A}}_{i-1}(\mathsf{pp}, \rho)$
2. Retrieve $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \mathit{\Pi}_i)$ from the internal state of $\mathcal{A}_{i-1}$
3. Parse $\mathit{\Pi}_i.\mathsf{v}_i$ to retrieve $z_{i-1}$ and $\omega_{i-1}$
4. Let $(\mathsf{v}_{i-1}, \mathsf{V}_{i-1}) \leftarrow \tilde{\mathcal{E}}_{i-1}(\mathsf{pp}, \rho)$.
5. Let $\mathit{\Pi}_{i-1} \leftarrow ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$
6. Output $((z_{i-1}, \ldots, z_{n-1}), (\omega_{i-1}, \ldots, \omega_{n-1}), \mathit{\Pi}_{i-1})$
```

We first reason that the output $(z_{i-1}, \ldots, z_{n-1})$, and $(\omega_{i-1}, \ldots, \omega_{n-1})$ are valid. By the inductive hypothesis, we already have that for all $j \in \{i+1, \ldots, n\}$,

$$z_j = F(z_{j-1}, \omega_{j-1}),$$

and that $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \mathit{\Pi}_i) = 1$ with the exception of $\mathsf{negl}(\lambda)$. Because $\mathcal{V}$ additionally checks that

$$\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i, Y_i) \tag{20}$$

by the construction of F_1' and the binding property of the hash function, we have

$$F(z_{i-1}, \omega_{i-1}) = z_i$$

with the exception of $\mathsf{negl}(\lambda)$. Next, we argue that $\mathit{\Pi}_{i-1}$ is valid. Because $(\mathsf{u}_i, \mathsf{v}_i)$ satisfies F', and $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ were retrieved from v_i, by the binding property of the hash function, and by Eq. (20), we have that

$$\begin{aligned} \mathsf{u}_{i-1}.\mathsf{x} &= \mathsf{hash}(\mathsf{vk}, i-1, z_0, z_{i-1}, \mathsf{U}_{i-1}, Y_{i-1}) \\ (\mathsf{u}_{i-1}.E, \mathsf{u}_{i-1}.s) &= (\mathsf{u}_\perp.E, 1) \end{aligned}$$

Additionally, in the case where $i = 1$, by the base case check of F_1', we have that $z_{i-1} = z_0$. Because $\tilde{\mathcal{E}}_{i-1}$ succeeds with the exception of $\mathsf{negl}(\lambda)$, we have that

$$\mathcal{V}(\mathsf{vk}, i-1, z_0, z_{i-1}, \mathit{\Pi}_{i-1}) = 1$$

with the exception of at most $\mathsf{negl}(\lambda)$. □