Tight State-Restoration Soundness in the Algebraic Group Model

Ashrujit Ghoshal and Stefano Tessaro

Paul G. Allen School of Computer Science & Engineering
University of Washington, Seattle, USA
{ashrujit,tessaro}@cs.washington.edu

Abstract. Most efficient zero-knowledge arguments lack a concrete security analysis, making parameter choices and efficiency comparisons challenging. This is even more true for non-interactive versions of these systems obtained via the Fiat-Shamir transform, for which the security guarantees generically derived from the interactive protocol are often too weak, even when assuming a random oracle.
This paper initiates the study of *state-restoration soundness* in the algebraic group model (AGM) of Fuchsbauer, Kiltz, and Loss (CRYPTO '18). This is a stronger notion of soundness for an interactive proof or argument which allows the prover to rewind the verifier, and which is tightly connected with the concrete soundness of the non-interactive argument obtained via the Fiat-Shamir transform.
We propose a general methodology to prove tight bounds on state-restoration soundness, and apply it to variants of Bulletproofs (Bootle et al, S&P '18) and Sonic (Maller et al., CCS '19). To the best of our knowledge, our analysis of Bulletproofs gives the *first* non-trivial concrete security analysis for a non-constant round argument combined with the Fiat-Shamir transform.

Keywords. Zero-knowledge proof systems, concrete security, Fiat-Shamir transform, Algebraic Group Model, state-restoration soundness.

1 Introduction

The last decade has seen zero-knowledge proof systems [1] gain enormous popularity in the design of efficient privacy-preserving systems. Their concrete efficiency is directly affected by the choice of a security parameter, yet *concrete security* analyses are rare and, as we explain below, hit upon technical barriers, even in ideal models (such as the random-oracle [2] or the generic-group models [3,4]). This has led to parameter choices not backed by proofs, and to efficiency comparisons across protocols with possibly incomparable levels of security. This paper addresses the question of narrowing this gap for protocols whose security can be analyzed in the Algebraic Group Model [5].

A CONCRETE EXAMPLE. It is convenient to start with an example to illustrate the challenges encountered in proving concrete security of proof systems. We focus on Bulletproofs [6], which are argument systems with applications across

the cryptocurrencies[1] and in verifiably deterministic signatures [9], which in turn optimize prior work [10]. The soundness[2] analysis (of their interactive version) is asymptotic, based on the hardness of the *discrete logarithm problem* (DLP). Even when instantiated from 256-bit elliptic curves, due to the absence of a tight, concrete, reduction, we have no formal guarantee on concrete security. Indeed, recent work [11] gives concrete soundness bounds in the generic-group model with somewhat unfavorable dependence on the size of the statement being proved, and no better analysis is known.

Even more importantly, existing bounds are for the *interactive* version of the protocol, but Bulletproofs are meant to be used *non-interactively* via the Fiat-Shamir (FS) transform [12]. However, the (folklore) analysis of the FS transform gives no useful guarantees: Namely, for a soundness bound ε on the *interactive* ZK proof system, the resulting NIZK has soundness $q^r\varepsilon$, where q is the number of random-oracle queries, and r is the number of challenges sent by the verifier. For Bulletproofs, we have $\varepsilon \geqslant 2^{-256}$ (this is the probability of merely *guessing* the discrete log), and if (say) $r = \Theta(\log(n)) \geqslant 16$, we only get security for (*at best*) $q \leqslant 2^{16}$ queries, which is clearly insufficient.

Overview of this paper. This paper studies the concrete security of succinct proof systems in the *algebraic group model* (AGM) [5], with the goal of developing (near-)exact security bounds. The AGM considers in particular *algebraic* provers that provide representations of group elements to the reduction (or to the extractor), and has been successful to study security in a variety of contexts. More specifically, this work is the first to look at *multi*-round *public-coin* protocols *and* their non-interactive version obtained via the Fiat-Shamir transform. For the latter, we aim for bounds with *linear* degradation in the number of random oracle queries q even for a large number of rounds r, as opposed to the q^r degradation obtained from naïve analyses. Prior work [5] has focused on the simpler case of linear-PCP based SNARKs [13], which are built from two-move interactive proofs and without the FS transform.

The soundness of non-interactive systems resulting from the FS transform is tightly related to the *state-restoration soundness* [14,15] of the underlying interactive protocol, where the cheating prover can *rewind* the verifier as it pleases, until it manages to complete a full accepting interaction with the verifier. No non-trivial bounds on state-restoration soundness are currently known on any non-constant round *argument*.

We propose a general framework to quantitatively study state-restoration version of *witness-extended emulation* (wee) [16,17] (which implies both state-restoration soundness and a proof-of-knowledge property) in the AGM. We then and apply it to three case studies, which include two variants of Bulletproofs, as well as Sonic [18]. These protocols have previously been analyzed only with respect to plain soundness in the interactive setting. The analysis of Bulletproofs

[1] In particular, Bulletproofs have been deployed in Monero [7] and Signal's upcoming MobileCoin [8].

[2] In this introduction, security is with respect to soundness – usually the analysis of zero-knowledge security is much more straightforward.

relies in particular on the Forking Lemma of Bootle *et al.* [10], which was only very recently made concrete [11]. We believe that our framework can be applied to a number of other protocols, such as Hyrax [19], Dory [20] or pairing-based instantiations of IOPs [21,22], and leave their analysis for future work.

Remark 1. We stress that our approach differs formally from prior and concurrent works (e.g., [18,22]) which use the AGM to give a heuristic validation of the security of a *component* of a protocol, which is then however assumed to satisfy extractability properties compatible with a standard-model proof (i.e., an AGM extractor is used as a standard-model extractor.) Here, we aim for full analyses in the AGM, and as we point out in our technical overview below, these approaches actually do not give a full-fledged proof in the AGM (beyond not giving a proof in the standard model either).

BULLETPROOFS. We apply our framework to two instantiations of Bulletproofs – the first is for *range proofs*, and the other is for general satisfiability of arithmetic circuits. For example, in the former, a prover shows in $O(\log n)$ rounds that for a given Pedersen commitment $C = g^v h^r$ in a cyclic group $\mathbb{G}$ of prime order p we have $v \in [0, 2^n)$. (Here, clearly, $2^n \leqslant p$.)

For the final non-interactive protocol obtained via the FS transform, our result implies that an (algebraic) t-time prover making q random-oracle queries can break security as a Proof of Knowledge (when properly formalized) with advantage roughly

$$\varepsilon(t, q) \leqslant O(qn/p) + \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(t) \ , \tag{1}$$

where $\mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(t)$ is the advantage of breaking the DLP within time t. In the generic group model, this is roughly $O(t^2/p)$, and this bound justifies the instantiation of Bulletproofs from a 256-bit curve. For arithmetic circuit satisfiability, we obtain a similar bound.

TIGHTNESS AND DISCUSSION. Assuming $\mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(t) \sim t^2/p$ (which is true in the generic group model), the above bound implies in particular that for most values of n,[3] the term $O(qn/p)$ is not leading. Still, we show that the dependence on n is necessary – in particular, we show that there exist n, p for which we can construct a cheating prover that can break soundness with probability $\Omega(qn/p)$, meaning that this part of the bound is tight. (Our argument can be extended to all bounds claimed in the paper.) Also, the term $\mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(t)$ is clearly necessary, given that breaking the DLP would directly give us an attack. This makes our bound essentially exact (up to small constants).

AGM AND COMPOSITION. A challenging aspect of our analysis is the difficulty of dealing with composition. The core of the Bulletproofs is indeed its $O(\log(n))$-round *inner-product argument*. In the standard model, and in the interactive case, it is not hard to reduce the security (as a proof of knowledge) of the full-fledged system using Bulletproofs to the analysis of the underlying inner-product argument, but it is not that clear how to do this generically in the

[3] For the circuit satisfiability version of our result, one should think of $n = 2^{20}$ and $p = 2^{256}$ as representative values.

AGM. In particular, in the AGM, the adversary provides representations of group elements to the reduction (or the extractor), and these are as a function of all priorly given group elements. The problem is that when analyzing a protocol in *isolation* (such as the inner-product argument) the bases to which elements are described are not necessarily the same as those that would be available to a cheating algebraic prover against the *full* protocol. This makes it hard to use an extractor for the inner-product argument in isolation as a sub-routine to obtain an extractor for a protocol using it. Also, because we consider state-restoration soundness, a sub-protocol can be initiated by a cheating prover several times, with several choices of these basis elements.

The downside of this is that our analyses are not modular, at least not at a level which considers sub-protocols are isolated building blocks – we give two different analyses for two different instantiations of Bulletproofs, and the shared modularity is at the algebraic level.

We discuss this further at the end of our technical overview below.

Sonic. As a second application, we study Sonic [18]. This is a constant-round protocol, and in particular with $3M + 2$ challenges for some constant $M \geqslant 1$. In this case, the folklore analysis of the FS transform can be used to obtain a non-trivial bound, incurring a multiplicative loss of q^{3M+2} from the soundness of the interactive version. Here, we want to show that this loss is not necessary and also obtain a bound which degrades linearly in q. Moreover, no concrete bound on the concrete soundness of Sonic was given in the interactive setting.

We ignore the stronger requirement of updatable witness-extended emulation because our pedagogical point here is that our framework can improve soundness even for constant-round protocols.

We also note that Sonic's proof already uses the AGM to justify security of the underlying polynomial commitment scheme, but follows a (heuristic) pattern described above where the resulting extractor is expected to behave as a standard-model one, and is used within a standard-model proof.

Adaptive vs non-adaptive soundness. It is important to understand that one can consider both *adaptive* and *non-adaptive* provers, where the former also chooses the *input* for which it attempts to provide a proof. Clearly, one expects adaptive provers to be harder to handle, but this is not necessarily true for *algebraic* provers – in particular, *if* the input contains group elements, the extractor can obtain useful information (and, possibly, directly extract) from their group representation. While this does not render the proof trivial at all, it turns out that for non-adaptive security, the proof is *even harder*. In this paper, we deal mostly with adaptive provers, but for the case of range proofs (where the inputs are commitments in a group), we also give a proof for non-adaptive security – the resulting bound is increased to the square root of the adaptive bound, due to our limited use of rewinding.

Related work: Proofs vs arguments. We clarify that state-restoration soundness has been studied for several forms of interactive *proofs* [14,15,23,24], also in its equivalent form of "round-by-round" soundness. Some proof systems satisfy it directly (such as those based on the sumcheck protocol [25]), whereas

any proof with non-trivial (plain) soundness can be amplified into one with sufficient stare-restoration soundness (e.g., with parallel repetition). This is because (similar to our statement about the Fiat-Shamir transform above) one can naïvely infer that a concrete soundness bound ε implies a state-restoration soundness bound $q^r\varepsilon$, where r is the number of challenges, and thus ε needs to be smaller than q^{-r}.

However, we do not know of any non-trivial bounds on state-restoration soundness for multi-round arguments based on computational assumptions (as opposed to, say, arguments in the ROM), and moreover, soundness amplification (e.g., [26,27,28,29]) does not reduce soundness beyond the largest negligible function, and this is insufficient to absorb the q^r loss.

Beyond the AGM. Our results are inherently based on online extraction, which is only meaningful in ideal models or using knowledge assumptions. One scenario where ideal models are inherently used is in the compilation of IOPs into NIZKs in the ROM via the BCS transform [14] – it is unclear whether our technique can be used to give tight state-restoration soundness bounds for systems such as Aurora [30] and STARK [31].

Concurrent Work. In a recently updated version of [32], Bünz *et. al.* analyse the soundness of the non-interactive inner-product argument of Bulletproofs in the AGM. We provide a brief comparison with their result in the full version [34], but note here that their analysis is asymptotic, and gives weaker concrete security (insufficient for instantiations on 256-bit curves) when made concrete.

1.1 Overview of our Techniques

We give a general framework to derive tight bounds on state-restoration soundness in the AGM. In fact, we will target the stronger notion of *witness-extended emulation* [16,17], which we adapt to state-restoration provers. Recall first that the main characteristic of the AGM is that it allows the reduction, or in our case the extractor, to access representations of group elements. A contribution of independent interest is to set up a formal framework to define extraction in the AGM.

Preface: Online Extraction in the AGM. In the AGM, the reduction (or an extractor) obtains *representations* of each group element in terms of all previously seen group elements. A useful feature of the AGM is that it often (but not always) allows us to achieve *online witness extraction*, as already observed in [5,33]. In other words, by looking at the representation of the group elements provided by the prover *in a single interaction*, the extractor is able to extract a witness, without the need of rewinding.

Online extraction however immediately appears to be very useful to tame the complexity of state-restoration provers. Indeed, one can visualize an interaction of an adversarial state-restoration prover $\mathcal{P}^*$ with the verifier V as defining an *execution tree*. In particular, $\mathcal{P}^*$ wins if it manages to create a path in the execution tree associated with an accepting (simple) transcript $\tau = (a_1, c_1, a_2, \ldots, c_r, a_{r+1})$,where $a_1, a_2, \ldots, a_{r+1}$ are $\mathcal{P}^*$'s messages, and $c_1, \ldots, c_r$

are the verifier's challenges. (We focus on public-coin protocols here.) Online extraction from a single transcript τ *directly* implies extraction here, because a witness can directly be extracted *locally* from the path τ (and the corresponding representations of group elements), disregarding what happened in the rest of the execution tree. In particular, the probability that $\mathcal{P}^*$ succeeds equals the probability that a witness is extracted. Without online extraction, we would have to use rewinding – but current techniques [10,11] do not seem to easily extend to state-restoration provers.

However, this only holds for *perfect* online extraction – in general, we may be able to generate transcripts which are accepting, but for which no witness can be extracted. This is typically because of two reasons:

- **Bad Challenges.** A bad choice of challenges may prevent witness extraction.
- **Violating an assumption.** A transcript is accepting, but the resulting interaction corresponds to a violation of some underlying assumption (i.e., one can extract a non-trivial discrete logarithm relation).

Our framework will exactly follow this pattern. For an r-challenge public-coin protocol, we identify bad challenges, i.e., for each $i \in [r]$, input x, and partial transcript $\tau' = (a_1, c_1, \ldots, a_{i-1}, c_{i-1}, a_i)$, we define a set of bad challenges c_i which would make extraction impossible. Crucially, these sets are defined according to a *simple interaction transcript* (i.e., not a state-restoration one) and can be defined according to the representation of group elements in the transcript so far. Then, given a transcript τ with no bad challenges, we show that:

- We can either extract a witness for x from τ (and the representations of the group elements in τ).
- We can use τ (and the representation of the group elements in terms of the public parameters) to break some underlying assumption.

To illustrate this, we give a non-trivial example next, which considers a simplified instance of the inner product argument at the core of Bulletproofs, but which already captures all subtleties of the model.

Inner-Product Argument of Bulletproofs. In the inner product argument the prover proves that a group element $P \in \mathbb{G}$ is a well-formed commitment to vectors $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n$ and their inner-product $\langle \mathbf{a}, \mathbf{b} \rangle$.[4] More precisely, the prover wants to prove to the verifier that $P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} u^{\langle \mathbf{a}, \mathbf{b} \rangle}$ where $\mathbf{g} \in \mathbb{G}^n, \mathbf{h} \in \mathbb{G}^n, u \in \mathbb{G}$ are independent generators of $\mathbb{G}$.

Here, we shall focus on the special case $n = 2$ first, and below discuss challenges in scaling our analysis up to any n. The prover first sends to the verifier group elements L, R where

$$L = g_2^{a_1} h_1^{b_2} u^{a_1 b_2} \ , \ R = g_1^{a_2} h_2^{b_1} u^{a_2 b_1} \ .$$

[4] We use boldface to denote vectors. For two vectors $\mathbf{a} = (a_1, \ldots, a_n), \mathbf{g} = (g_1, \ldots, g_n)$, we use $\mathbf{g}^{\mathbf{a}}$ to denote $\prod_{i=1}^{n} g_i^{a_i}$.

The verifier samples x uniformly at random from $\mathbb{Z}_p^*$ and sends it to the prover. We then define

$$P' = L^{x^2} P R^{x^{-2}} \ , \ g' = g_1^{x^{-1}} g_2^{x} \ , \ h' = h_1^{x} h_2^{x^{-1}} \ .$$

The prover sends $a' = a_1 x + a_2 x^{-1}$ and $b' = b_1 x^{-1} + b_2 x$ to the verifier, which in turns accepts if and only if

$$P' = (g')^{a'} (h')^{b'} u^{a'b'} \ .$$

EXTRACTION FOR $n = 2$. For this discussion, we focus in particular on the notion of *adaptive* soundness – i.e., the prover provides P along with its representation, i.e, we get $\mathbf{a}' = (p_{g_1}, p_{g_2})$, $\mathbf{b}' = (p_{h_1}, p_{h_2})$ and p_u such that $P = \mathbf{g}^{\mathbf{a}'} \mathbf{h}^{\mathbf{b}'} u^{p_u}$. At first, it looks like we are done – after all, we can just check whether $\langle \mathbf{a}', \mathbf{b}' \rangle = p_u$, and if so, output $(\mathbf{a}', \mathbf{b}')$ as our witness. Unfortunately, things are not *that* simple – we need to ensure that no accepting transcript $\tau = ((L, R), x, (a', b'))$, i.e., such that $P' = (g')^{a'} (h')^{b'} u^{a'b'}$, is ever produced if $\langle \mathbf{a}', \mathbf{b}' \rangle \neq p_u$, for otherwise our naïve extraction would fail.

To this end, we will prove that if the cheating prover can produce an accepting interaction such while $\langle \mathbf{a}', \mathbf{b}' \rangle \neq p_u$, then we can solve the discrete logarithm problem in the group $\mathbb{G}$. We construct an adversary $\mathcal{A}$ that takes as inputs g_1, g_2, h_1, h_2, u and attempts to return a non-trivial discrete logarithm relation between them. (Breaking this is *tightly* equivalent to breaking the discrete logarithm problem.) Concretely, the adversary $\mathcal{A}$ gives g_1, g_2, h_1, h_2, u as input to the cheating prover $\mathcal{P}$, which first returns an adaptively chosen input $P \in \mathbb{G}$, along with is algebraic representation

$$P = g_1^{p_{g_1}} g_2^{p_{g_2}} h_1^{p_{h_1}} h_2^{p_{h_2}} u^{p_u} \ .$$

The adversary then simulates the execution of $\mathcal{P}$ with a honest verifier further, and assumes it generates an accepting transcript $\tau = ((L, R), x, (a', b'))$ – this transcript contains the representations of L, R such that $L = g_1^{l_{g_1}} g_2^{l_{g_2}} h_1^{l_{h_1}} h_2^{l_{h_2}} u^{l_u}$ and $R = g_1^{r_{g_1}} g_2^{r_{g_2}} h_1^{r_{h_1}} h_2^{r_{h_2}} u^{r_u}$ and since it is an accepting transcript we have

$$L^{x^2} P R^{x^{-2}} = g_1^{x^{-1}a'} g_2^{x^1 a'} h_1^{x^1 b'} h_2^{x^{-1} b'} u^{a'b'} \ .$$

We can plug in the representations of L, R into the equality and obtain values $e_{g_1}, e_{g_2}, e_{h_1}, e_{h_2}, e_u$ such that

$$g_1^{e_{g_1}} g_2^{e_{g_2}} h_1^{e_{h_1}} h_2^{e_{h_2}} u^{e_u} = 1 \ . \tag{2}$$

For example $e_{g_1} = x^{-1}a' - l_{g_1} x^2 - r_{g_1} x^{-2} - p_{g_1}$ and $e_u = a'b' - l_u x^2 - r_u x^{-2} - p_u$.

The adversary $\mathcal{A}$ then simply outputs $(e_{g_1}, e_{g_2}, e_{h_1}, e_{h_2}, e_u)$ – it has found a non-trivial discrete logarithm relation if $(e_{g_1}, e_{g_2}, e_{h_1}, e_{h_2}, e_u) \neq (0, 0, 0, 0, 0)$, which we next show happens with very high probability if $p_u \neq p_{g_1} p_{h_1} + p_{g_2} p_{h_2}$.

Suppose $(e_{g_1}, e_{g_2}, e_{h_1}, e_{h_2}, e_u) = (0, 0, 0, 0, 0)$. From $e_{g_1} = 0$, we have that $x^{-1}a' - l_{g_1} x^2 - r_{g_1} x^{-2} - p_{g_1} = 0$. Since $x \neq 0$, we get that $a' = l_{g_1} x^3 + r_{g_1} x^{-1} +$

$p_{g_1} x$. Similarly from $e_{g_2} = 0$, we would get $a' = l_{g_2} x + p_{g_2} x^{-1} + r_{g_2} x^{-3}$. With high probability over the choice of x's, by the Schwartz-Zippel Lemma, we can infer by equating both right-hand sides that

$$a' = x p_{g_1} + x^{-1} p_{g_2} \ .$$

Similarly, from $e_{h_1} = 0$ and $e_{h_2} = 0$, we obtain that

$$b' = x^{-1} p_{h_1} + x p_{h_2}$$

for most x's. Finally, from $e_u = 0$, we similarly learn that

$$a'b' = x^2 l_u + p_u + x^{-2} r_u \ .$$

Hence from the above

$$x^2 l_u + p_u + x^{-2} r_u = p_{g_1} p_{h_1} + p_{g_2} p_{h_2} + p_{g_1} p_{h_2} x^2 + p_{g_2} p_{h_1} x^{-2} \ .$$

Since we have that $p_{g_1} p_{h_1} + p_{g_2} p_{h_2} \neq p_u$, the above equality holds with very small probability over the choice of x's.

Hence we have shown that $(e_{g_1}, e_{g_2}, e_{h_1}, e_{h_2}, e_u) = (0, 0, 0, 0, 0)$ with very small probability. Therefore $\mathcal{A}$ succeeds with high probability.

Non-adaptive security. The above proof exploits the fact that the prover *provides* a representation of P – this corresponds to the case of an *adaptive* prover. But there are scenarios where the prover may be non-adaptive and not be able to do that – for example, the input P has been generated by *another* party, and the prover tries to prove knowledge with respect to this P. It turns out that in this case, one needs a different proof. In fact, one *could* give an extraction strategy which does not require knowing an initial representation for P, but it is then hard to give a reduction to the discrete logarithm problem to show correctness.

We stress that non-adaptive provers and adaptive provers are equivalent in many applications – they only differ when the input includes group elements. We give a formalization and a case study (for Bulletproofs range proofs) in the full version [34].. There, we can actually give a reduction to the discrete logarithm problem (to bound the probability of failing to extract), but this requires rewinding *once* – this allows us to prove a bound which is the square root of the bound for adaptive provers.

The recursive protocol for $n = 4$. Scaling the protocol to an arbitrary n proceeds via recursion. For concreteness, let us focus on the case $n = 4$. The prover first sends to the verifier group elements L, R where

$$L = g_3^{a_1} g_4^{a_2} h_1^{b_3} h_2^{b_4} u^{a_1 b_3 + a_2 b_4} \ , \ R = g_1^{a_3} g_2^{a_4} h_3^{b_1} h_4^{b_2} u^{a_3 b_1 + a_4 b_2} \ .$$

The verifier samples x uniformly at random from $\mathbb{Z}_p^*$ and sends it to the prover. The prover and the verifier both compute

$$P' = L^{x^2} P R^{x^{-2}} \ , \ g_1' = g_1^{x^{-1}} g_3^x \ , \ g_2' = g_2^{x^{-1}} g_4^x \ , \ h_1' = h_1^x h_3^{x^{-1}} \ , \ h_2' = h_2^x h_4^{x^{-1}} \ .$$

The prover also computes $a'_1 = a_1x + a_3x^{-1}$, $a'_2 = a_2x + a_4x^{-1}$, $b'_1 = b_1x^{-1} + b_3x$ and $b'_2 = b_2x^{-1} + b_4x$. Observe that $P' = (g'_1)^{a'_1}(g'_2)^{a'_2}(h'_1)^{b'_1}(h'_3)^{b'_2}u^{a'_1b'_1+a'_2b'_2}$. Now, the prover and the verifier engage, recursively, in the protocol for $n = 2$ with inputs $(g'_1, g'_2), (h'_1, h'_2), u, P', (a'_1, a'_2), (b'_1, b'_2)$. The difficulty in analyzing this is that we would like our proof strategy to be recursive, i.e., given we analyzed the protocol for n secure, we can now infer that the one for $2n$ also is secure. This will not be so direct, unfortunately. One major technical issue is for example that the recursive call uses different generators than the ones used for the calling protocol – in our case, here, $(g'_1, g'_2), (h'_1, h'_2)$ – however, when looking at the combined protocol in the AGM, all element representations would be with respect to the generators $g_1, \ldots, g_4, h_1, \ldots, h_4$, and this makes it difficult to directly recycle the above analysis.

The challenges with composition. The inability to leverage recursion to simplify the approach from the previous paragraph is not an isolated incident. We note that a non-trivial aspect of our analyses is due to the lack of easy composition properties in the AGM. In particular, we encounter the following problem – if we have a protocol Π' (e.g., the inner-product argument) which is used as a sub-protocol for Π (a Bulletproofs range proof), and we prove extractability for Π', it is not clear we can infer extractability for Π in a modular way by just calling the extractor for Π'. This is because a stand-alone analysis of Π' may assume group elements output by a malicious prover $\mathcal{P}'$ are represented with respect to some set of basis elements – say, the generators $g_1, \ldots, g_n, h_1, \ldots, h_n, u$ in the concrete example of inner-product argument described above. However, when Π' is used within Π, the generators of the inner-product argument are functions of *different group* elements. When studying a prover $\mathcal{P}$ attacking Π, then, representations of group elements are with respect to this different set of group elements, and this makes it hard to use an extractor for Π' directly, as it assumes different representations.

This is a problem we encounter in our analyses, and which prevents us from abstracting a theorem for the inner-product argument which we could use, in a plug-and-play way, to imply security of higher-level protocols using it. The flip side is that this lack of composability also comes to our advantage – our extractors will in fact not even need to extract anything from the transcript of an accepting execution of the inner-product argument, but only use the fact that it is accepting to infer correctness of the extracted value.

The issue with prior AGM analyses. Composition issues seemingly affect existing analyses of proof systems in the literature (e.g., [18,22]), whenever some components are analyzed in the AGM (typically, a polynomial commitment scheme), but the overall proof is expressed in the standard model. As far as we can tell, unlike this work, one cannot directly extract a full AGM analysis from these works – let us elaborate on this.

Obviously, from a purely formal perspective, the standard model and the algebraic group model cannot be quite mixed, as in particular the AGM extractor for the component cannot be used in the standard model – the only formally correct way to interpret the analysis is as *fully* in the AGM, but part of the analysis

does not leverage the full power of the model, and is effectively a standard-model reduction. Yet, in order for composition to be meaningful, it is important to verify that the basis elements assumed in the AGM analysis of the components are the same available to a prover attacking the complete protocol. While we cannot claim any issues (in fact, we give an analysis of Sonic in this paper with a concrete bound), it does appear that all existing works do not attempt to provide a formal composition – they use the existence of an AGM extractor as a heuristic validation for the existence of a standard-model extractor, rather than making formally correct use as an AGM extractor within an AGM proof. Making this composition sound is potentially non-trivial. Having said this, for pairing-based polynomial commitment schemes, the basis elements are generally the same, and thus this can likely be made rigorous fairly easily (unlike the case of inner-product arguments).

2 Preliminaries

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ represent the set of all natural numbers and let $\mathbb{N}^+ = \mathbb{N} \backslash \{0\}$. For $N \in \mathbb{N}^+$, let $[N] = \{1, \ldots, N\}$. We use $\mathsf{Pr}[\mathsf{G}]$ to denote the probability that the game G returns `true`. Let $\mathbb{G}$ be a cyclic group of prime order p with identity 1 and let $\mathbb{G}^* = \mathbb{G} \backslash \{1\}$ be the set of its generators. We use boldface to denote a vector, e.g., $\mathbf{g} \in \mathbb{G}^n$ is a vector of n group elements with its i^{th} element being g_i, i.e., $\mathbf{g} = (g_1, \ldots, g_n)$. For two vectors $\mathbf{a} = (a_1, \ldots, a_n), \mathbf{g} = (g_1, \ldots, g_n)$, we use $\mathbf{g}^{\mathbf{a}}$ to denote $\prod_{i=1}^{n} g_i^{a_i}$. We use python notation to denote slices of vectors:

$$\mathbf{g}_{[:l]} = (g_1, \ldots, g_l) \in \mathbb{G}^l \ , \ \mathbf{g}_{[l:]} = (g_{l+1}, \ldots, g_n) \in \mathbb{G}^{n-l} \ .$$

For $z \in \mathbb{Z}_p^*$, we use $\mathbf{z}^n$ to denote the vector $(1, z, z^2, \ldots, z^{n-1})$. Similarly, we use $\mathbf{z}^{-n}$ to denote the vector $(1, z^{-1}, z^{-2}, \ldots, z^{-n+1})$. If Z is a variable, $\mathbf{Z}^n$ represents the vector $(1, Z, Z^2, \ldots, Z^{n-1})$. Our vectors are indexed starting from 1, so $\mathbf{z}^{n+1}_{[1:]}$ is the vector $(z, z^2, \ldots, z^n)$. The operator $\circ$ denotes the Hadamard product of two vectors, i.e., $\mathbf{a} = (a_1, \ldots, a_n), \mathbf{b} = (b_1, \ldots, b_n), \mathbf{a} \circ \mathbf{b} = (a_1 b_1, \ldots, a_n b_n)$. We use capitalized boldface letters to denote matrices, e.g., $\mathbf{W} \in \mathbb{Z}_p^{n \times m}$ is a matrix with n rows and m columns.

We denote the inner product of two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n$ using $\langle \mathbf{a}, \mathbf{b} \rangle$. We also define vector polynomials, e.g., $f(X) = \sum_{i=0}^{d} \mathbf{f}_i X^i$, where each coefficient $\mathbf{f}_i$ is a vector in $\mathbb{Z}_p^n$.

The function $\mathsf{bit}(k, i, t)$ returns the bit k_i where $(k_1, \ldots, k_t)$ is the t-bit representation of k.

Schwartz-Zippel Lemma. The polynomial ring in variables $X_1, \ldots, X_n$ over the field $\mathbb{F}$ is denoted by $\mathbb{F}[X_1, \ldots, X_n]$.

Lemma 1 (Schwartz-Zippel Lemma). *Let $\mathbb{F}$ be a finite field and let $f \in \mathbb{F}[X_1, \ldots, X_n]$ be a non-zero n variate polynomial with maximum degree d. Let S be a subset of $\mathbb{F}$. Then $\mathsf{Pr}[f(x_1, \ldots, x_n) = 0] \leqslant d/|S|$, where the probability is over the choice of $x_1, \ldots, x_n$ according to $x_i \leftarrow\!\!\$\ S$.*

Game $\mathsf{G}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{A},\lambda)$:	**Game** $\mathsf{G}^{\mathsf{dl-rel}}_{\mathbb{G},n}(\mathcal{A},\lambda)$:	**Game** $\mathsf{G}^{q\text{-}\mathsf{dl}}_{\mathbb{G}}(\mathcal{A},\lambda)$:
$g \leftarrow\!\!\$\ \mathbb{G}_\lambda{}^*$; $h \leftarrow\!\!\$\ \mathbb{G}_\lambda$	$g_1,\ldots,g_n \leftarrow\!\!\$\ \mathbb{G}_\lambda$	$g \leftarrow\!\!\$\ \mathbb{G}_\lambda{}^*$
$a \leftarrow\!\!\$\ \mathcal{A}_\lambda(g,h)$	$(a_1,\ldots,a_n) \leftarrow\!\!\$\ \mathcal{A}_\lambda(g_1,\ldots,g_n)$	$x \leftarrow\!\!\$\ \mathbb{Z}_{p(\lambda)}$
Return $(g^a = h)$	Return $(\prod_{i=1}^{n} g_i^{a_i} = 1 \wedge (a_1,\ldots,a_n) \neq \mathbf{0}^n)$	$x' \leftarrow\!\!\$\ \mathcal{A}_\lambda(\{g^{x^d}\}_{d=-q}^{q})$
		Return $(x = x')$

Fig. 1. The games used to define the advantage of a non-uniform adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}^+}$ against the discrete logarithm problem, the discrete logarithm relation problem and the q-DLOG problem in a family of cyclic groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ with prime order order $p = p(\lambda)$. The set $\mathbb{G}_\lambda{}^*$ is the set of generators of $\mathbb{G}_\lambda$.

In particular if p is a prime and $f \in \mathbb{Z}_p[X]$ is a polynomial of degree d and x is sampled uniformly at random from $\mathbb{Z}_p^*$, then $\mathsf{Pr}\left[f(x) = 0\right] \leqslant d/(p-1)$. Further this implies that if $g(X) = f(X)/X^i$ for $i \in \mathbb{N}$ and x is sampled uniformly at random from $\mathbb{Z}_p^*$, then $\mathsf{Pr}\left[g(x) = 0\right] = \mathsf{Pr}\left[f(x) = 0\right] \leqslant d/(p-1)$.

The Discrete logarithm problem. The game $\mathsf{G}^{\mathsf{dl}}_{\mathbb{G}}$ in Figure 1 is used for is used for defining the advantage of a non-uniform adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}^+}$ against the discrete logarithm problem in a family of cyclic groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ of prime order $p = p(\lambda)$ with identity 1 and set of generators $\mathbb{G}^* = \{\mathbb{G}^*_\lambda\}_{\lambda\in\mathbb{N}^+} = \{\mathbb{G}_\lambda\backslash\{1\}\}_{\lambda\in\mathbb{N}^+}$. We define $\mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{A},\lambda) = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{A},\lambda)\right]$.

The Discrete logarithm relation problem. The game $\mathsf{G}^{\mathsf{dl-rel}}_{\mathbb{G},n}$ in Figure 1 is used for defining the advantage of a non-uniform adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}^+}$ against the discrete logarithm relation problem in a family of cyclic groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$. We define $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}^+}$ as $\mathsf{Adv}^{\mathsf{dl-rel}}_{\mathbb{G},n}(\mathcal{A},\lambda) = \mathsf{Pr}\left[\mathsf{G}^{\mathsf{dl-rel}}_{\mathbb{G},n}(\mathcal{A},\lambda)\right]$.The following lemma shows that hardness of the discrete logarithm relation problem in $\mathbb{G}$ is tightly implied by the hardness of discrete logarithm problem in a family of cyclic groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$.

Lemma 2. *Let $n \in \mathbb{N}^+$. Let $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ be a family of cyclic groups with order $p = p(\lambda)$. For every non-uniform adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}^+}$ there exists a non-uniform adversary $\mathcal{B} = \{\mathcal{B}_\lambda\}_{\lambda\in\mathbb{N}^+}$ such that for all $\lambda \in \mathbb{N}^+$, $\mathsf{Adv}^{\mathsf{dl-rel}}_{\mathbb{G},n}(\mathcal{A},\lambda) \leqslant \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{B},\lambda) + 1/p$. Moreover, $\mathcal{B}$ is nearly as efficient as $\mathcal{A}$.*

We refer the reader to [11] for a proof of this lemma.

The q-DLOG problem. The game $\mathsf{G}^{q\text{-}\mathsf{dl}}_{\mathbb{G}}$ in Figure 1 is used for defining the advantage of a non-uniform adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}^+}$ against the q-DLOG problem in a family of groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$. We define $\mathsf{Adv}^{q\text{-}\mathsf{dl}}_{\mathbb{G}}(\mathcal{A},\lambda) = \mathsf{Pr}\left[\mathsf{G}^{q\text{-}\mathsf{dl}}_{\mathbb{G}}(\mathcal{A},\lambda)\right]$.

We note that there are other problems known as q-DLOG which are not equivalent to the one we use here. We use the version stated above because it was the version used in the analysis of Sonic [18] which we analyse in this paper.

Game $\mathsf{SRS}^{\mathcal{P}}_{\mathsf{IP}}(\lambda)$:	**Oracle** $\mathbf{O}_{\mathrm{ext}}(\tau = (a_1, c_1, \ldots, a_{i-1}, c_{i-1}), a_i)$:
$\mathsf{win} \leftarrow \mathtt{false}$; $\mathsf{tr} \leftarrow \varepsilon$	If $\tau \in \mathsf{tr}$ then
$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{IP.Setup}(1^\lambda)$	If $i \leqslant r$ then
$(x, \mathsf{st}_{\mathcal{P}}) \leftarrow\!\!\$\ \mathcal{P}_\lambda(\mathsf{pp})$	$c_i \leftarrow\!\!\$\ \mathsf{Ch}_i$; $\mathsf{tr} \leftarrow \mathsf{tr} \parallel (\tau, a_i, c_i)$; Return c_i
Run $\mathcal{P}^{\mathbf{O}_{\mathrm{ext}}}_\lambda(\mathsf{st}_{\mathcal{P}})$	Else if $i = r + 1$ then
Return win	$d \leftarrow \mathsf{IP.V}(\mathsf{pp}, x, (\tau, a_i))$; $\mathsf{tr} \leftarrow \mathsf{tr} \parallel (\tau, a_i)$
	If $d = 1$ then $\mathsf{win} \leftarrow \mathtt{true}$
	Return d
	Return $\perp$

Fig. 2. Definition of state-restoration soundness. The game SRS defines state-restoration soundness for a non-uniform prover $\mathcal{P}$ and a public-coin interactive proof IP. Here, IP has $r = r(\lambda)$ challenges and the i-th challenge is sampled from Ch_i.

3 Interactive Proofs and State-restoration Soundness

We introduce our formalism for handling interactive proofs and arguments, which is particularly geared towards understanding their concrete state-restoration soundness.

Interactive proofs. An *interactive proof* [1] IP is a triple of algorithms: (1) the *setup algorithm* $\mathsf{IP.Setup}$ which generates the public parameters pp, (2) the *prover* $\mathsf{IP.P}$ and (3) the *verifier* $\mathsf{IP.V}$. In particular, the prover and the verifier are interactive machines which define a two-party protocol, where the prover does not produce any output, and the verifier outputs a decision bit $d \in \{0,1\}$. We let $\langle \mathsf{IP.P}(x), \mathsf{IP.V}(y) \rangle$ denote the algorithm which runs an execution of the prover and the verifier on inputs x and y, respectively, and outputs the verifier's decision bit. We say that IP is *public coin* if all messages sent from $\mathsf{IP.V}$ to $\mathsf{IP.P}$ are fresh random values from some understood set (which we refer to as *challenges*).

Completeness. A *relation* R is (without loss of generality) a subset of $\{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$. We denote a relation R that uses specified public parameters pp, instance x and witness w as $\{(\mathsf{pp}, x, w) : f_R(\mathsf{pp}, x, w)\}$ where $f_R(\mathsf{pp}, x, w)$ is a function that returns $\mathtt{true}$ if $(\mathsf{pp}, x, w) \in R$ and $\mathtt{false}$ otherwise. For every $\lambda \in \mathbb{N}^+$ and every $\mathcal{A}$, define the following experiment:

$$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{IP.Setup}(1^\lambda)\ ,\quad (x, w) \leftarrow\!\!\$\ \mathcal{A}(\mathsf{pp})\ ,\quad d \leftarrow\!\!\$\ \langle \mathsf{IP.P}(\mathsf{pp}, x, w), \mathsf{IP.V}(\mathsf{pp}, x) \rangle\ .$$

Then, we say that IP is an interactive proof for the relation R if for all $\mathcal{A}$ and all $\lambda \in \mathbb{N}^+$, in the above experiment the event $(d = 1) \vee ((\mathsf{pp}, x, w) \notin R)$ holds with probability one.

State-restoration soundness. We target a stronger notion of soundness – *state-restoration soundness* (SRS) [14,15] – which (as we show below) tightly reduces to the soundness of the non-interactive proof obtained via the Fiat-Shamir transform. The SRS security game allows the cheating prover to *rewind* the verifier as it pleases, and wins if and only if it manages to produce *some* accepting interaction. We only consider an $r(\lambda)$-challenge *public-coin* interactive proof IP, and consider the case where challenges are drawn uniformly from

some sets $\mathsf{Ch}_1, \ldots, \mathsf{Ch}_r$. We also assume that the verifier is described by an algorithm which given pp, x, and a *transcript* $\tau = (a_1, c_1, \ldots, a_r, c_r, a_{r+1})$, outputs a decision bit $d \in \{0,1\}$. We overload notation and write $\mathsf{IP.V}(\mathsf{pp}, x, \tau)$ for this output.

Our definition considers a game $\mathsf{SRS}^{\mathcal{P}}_{\mathsf{IP}}(\lambda)$ (which is formalized in Figure 2) that involves a non-uniform cheating prover $\mathcal{P} = \{\mathcal{P}_\lambda\}_{\lambda\in\mathbb{N}}$. (Henceforth, whenever we have any non-uniform adversary $\mathcal{A}$, it is understood $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ – we shall not specify this explicitly). The prover is initially responsible for generating the input x on which it attempts to convince the verifier on *some* execution. Its rewinding access to the verifier is ensured by an oracle $\mathbf{O}_{\text{ext}}$, to which it has access. Roughly speaking, the oracle allows the prover to build an *execution tree*, which is extended with each query to it by the prover. This execution tree can be inferred from tr, which sequentially logs all (valid) queries to $\mathbf{O}_{\text{ext}}$ by the prover. For a partial transcript τ', we write $\tau' \in \mathsf{tr}$ to mean that a partial execution corresponding to τ' can be inferred from tr.

We then associate the probability of winning the game with the srs *advantage metric*, $\mathsf{Adv}^{\mathsf{srs}}_{\mathsf{IP}}(\mathcal{P}, \lambda) = \mathsf{Pr}\left[\mathsf{SRS}^{\mathsf{IP}}_{\mathcal{P}}(\lambda)\right]$.For notational convenience, we do not restrict the input x not to have a witness. Therefore, if IP is an interactive proof for a relation R, we cannot hope to show that $\mathsf{Adv}^{\mathsf{srs}}_{\mathsf{IP}}(\mathcal{P}, \lambda)$ is small *for all* $\mathcal{P}$. Clearly, if $\mathcal{P}$ outputs (x, a) such that $(\mathsf{pp}, x, a) \in R$, then a is a witness and $\mathcal{P}$ can simply (honestly) convince the verifier. The classical notion of state-restoration soundness is recovered by only considering $\mathcal{P}$'s which output x such that $(\mathsf{pp}, x, w) \notin R$ for any w.

4 Proofs of Knowledge in the AGM

The Algebraic Group Model. We start here with a brief review of the AGM [5]. For an understood group $\mathbb{G}$ with prime order p, an *algebraic* algorithm $\mathcal{A}_{\mathsf{alg}}$ is an interactive algorithm whose inputs and outputs are made of distinct group elements and strings. Furthermore, each (encoding) of a group element X output by $\mathcal{A}_{\mathsf{alg}}$ is accompanied by a *representation* $(x_{A_1}, x_{A_2}, \ldots, x_{A_k}) \in \mathbb{Z}_p^k$ such that $X = \prod_{i=1}^{k} A_i^{x_{A_i}}$, where $A_1, \ldots, A_k$ are all group elements previously input *and* output by $\mathcal{A}_{\mathsf{alg}}$. Generally, we write $[X]$ for a group element X *enhanced* with its representation, e.g.,$[X] = (X, x_{A_1}, x_{A_2}, \ldots, x_{A_k})$. In particular, when we use a group element X output by $\mathcal{A}_{\mathsf{alg}}$, e.g. it is *input* to a reduction or used in a cryptographic game, we write $[X]$ to make explicit that the representation is available, whereas write X only when the representation is omitted. The notation extends to a mix of group elements and strings a – $[a]$ enhances each group element with its representation.

Defining AGM extraction. We formalize a notion of proof-of-knowledge (PoK) security in the AGM, following the lines of witness-extended emulation [16,17], which we extend to provers that can rewind the verifier.

We will be interested in cases where the AGM allows for online extraction, i.e., the additional group representations will allow for extraction without rewinding

Game $\mathsf{WEE\text{-}1}_{\mathsf{IP}}^{\mathcal{P}_{\mathsf{alg}},\mathcal{D}}(\lambda)$:
$\mathsf{tr} \leftarrow \varepsilon$
$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{IP.Setup}(1^\lambda)$
$([x], \mathsf{st}_{\mathcal{P}}) \leftarrow\!\!\$\ \mathcal{P}_{\mathsf{alg},\lambda}(\mathsf{pp})$
Run $\mathcal{P}_{\mathsf{alg},\lambda}^{\mathbf{O}^1_{\mathrm{ext}}}(\mathsf{st}_{\mathcal{P}})$
$b \leftarrow\!\!\$\ \mathcal{D}(\mathsf{tr})$
Return $(b = 1)$

Game $\mathsf{WEE\text{-}0}_{\mathsf{IP},R}^{\mathcal{E},\mathcal{P}_{\mathsf{alg}},\mathcal{D}}(\lambda)$:
$\mathsf{tr} \leftarrow \varepsilon$
$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{IP.Setup}(1^\lambda)$
$([x], \mathsf{st}_{\mathcal{P}}) \leftarrow\!\!\$\ \mathcal{P}_{\mathsf{alg},\lambda}(\mathsf{pp})$
$\mathsf{st}_{\mathcal{E}} \leftarrow (1^\lambda, \mathsf{pp}, [x])$
Run $\mathcal{P}_{\mathsf{alg},\lambda}^{\mathbf{O}^0_{\mathrm{ext}}}(\mathsf{st}_{\mathcal{P}})$
$w \leftarrow\!\!\$\ \mathcal{E}(\mathsf{st}_{\mathcal{E}}, \bot)$
$b \leftarrow\!\!\$\ \mathcal{D}(\mathsf{tr})$
Return $(b = 1) \ \wedge$
$(\mathsf{Acc}(\mathsf{tr}) \Rightarrow (\mathsf{pp}, x, w) \in R)$

Oracle $\mathbf{O}^1_{\mathrm{ext}}(\tau = (a_1, c_1, \ldots, a_{i-1}, c_{i-1}), a_i)$:
If $\tau \in \mathsf{tr}$ then
  If $i \leqslant r$ then
    $c_i \leftarrow\!\!\$\ \mathsf{Ch}_i$; $\mathsf{tr} \leftarrow \mathsf{tr} \parallel (\tau, a_i, c_i)$; return c_i
  Else if $i = r + 1$ then
    $d \leftarrow \mathsf{IP.V}(\mathsf{pp}, x, \tau \parallel a_i)$
    Return d
Return $\bot$

Oracle $\mathbf{O}^0_{\mathrm{ext}}(\tau = (a_1, c_1, \ldots, a_{i-1}, c_{i-1}), a_i)$:
If $\tau \in \mathsf{tr}$ then
  If $i \leqslant r$ then
    $(\mathsf{resp}, \mathsf{st}_{\mathcal{E}}) \leftarrow\!\!\$\ \mathcal{E}(\mathsf{st}_{\mathcal{E}}, [(\tau, a_i)])$
    $\mathsf{tr} \leftarrow \mathsf{tr} \parallel (\tau, a_i, \mathsf{resp})$
    Return resp
  Else if $i = r + 1$ then
    $d \leftarrow \mathsf{IP.V}(\mathsf{pp}, x, \tau \parallel a_i)$
    Return d
Return $\bot$

Fig. 3. Definition of online srs-wee security in the AGM. The games $\mathsf{WEE\text{-}1}, \mathsf{WEE\text{-}0}$ define online srs-wee security in the AGM for a non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$, a distinguisher $\mathcal{D}$, an extractor $\mathcal{E}$ and a public-coin interactive proof IP. We assume here that IP has $r = r(\lambda)$ challenges and the i-th challenge is sampled from Ch_i.

the prover. We target an adaptive notion of security, where the input is generated by the adversarial prover *itself*, depending on the public parameters pp, and can contain group elements.

Online srs-wee security. The definition consists of two games – denoted $\mathsf{WEE\text{-}1}_{\mathsf{IP}}^{\mathcal{P}_{\mathsf{alg}},\mathcal{D}}$ and $\mathsf{WEE\text{-}0}_{\mathsf{IP},R}^{\mathcal{E},\mathcal{P}_{\mathsf{alg}},\mathcal{D}}$, and described in Figure 3. The former captures the real game, lets our prover $\mathcal{P} = \{\mathcal{P}_\lambda\}_{\lambda\in\mathbb{N}}$ interact with an oracle $\mathbf{O}^1_{\mathrm{ext}}$ as in the state-restoration soundness game defined above, which additionally stores a transcript tr. The latter is finally given to a *distinguisher* $\mathcal{D}$ which outputs a decision bit. In contrast, the *ideal* game delegates the role of answering $\mathcal{P}$'s oracle queries to a (stateful) extractor $\mathcal{E}$. The extractor, at the end of the execution, also outputs a witness candidate for w. The extractor in particular exploits here the fact that $\mathcal{P}$ is algebraic by learning the representation of every input to the oracle $\mathbf{O}^0_{\mathrm{ext}}$. (This representation can be thought, without loss of generality, as being in terms of all group elements contained in pp.) Here, the final output of the game is not merely $\mathcal{D}$'s decision bit – should the latter output 1, the output of the game is `true` only if additionally the extracted witness is correct assuming the interaction with $\mathbf{O}^0_{\mathrm{ext}}$ resulted in an accepting execution – a condition we capture via the predicate $\mathsf{Acc}(\mathsf{tr})$.

For an interactive proof IP and an associated relation R, non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$, a distinguisher $\mathcal{D}$, and an extractor $\mathcal{E}$, we define

$$\mathsf{Adv}^{\mathsf{sr\text{-}wee}}_{\mathsf{IP},R}(\mathcal{P}_{\mathsf{alg}},\mathcal{D},\mathcal{E},\lambda) = \mathsf{Pr}\left[\mathsf{WEE\text{-}1}^{\mathcal{P}_{\mathsf{alg}},\mathcal{D}}_{\mathsf{IP}}(\lambda)\right] - \mathsf{Pr}\left[\mathsf{WEE\text{-}0}^{\mathcal{E},\mathcal{P}_{\mathsf{alg}},\mathcal{D}}_{\mathsf{IP},R}(\lambda)\right] . \quad (3)$$

One can consider also scenarios where the prover may be non-adaptive – for example, the input has been generated by *another* party, and the prover tries to prove knowledge with respect to this input. For this reason, introduce the notion of non-adaptive srs-wee in the full version [34].

4.1 The Basic Framework

We develop a general framework that we will use, via Theorem 1, to derive concrete AGM bounds on srs-wee security. Our goal, in particular, is to give conditions on *single* path executions – i.e., executions not involving any rewinding of the verifier by the prover, which could be seen as root-to-leaf paths in an execution tree generated by the interaction of a state-restoration prover.

Transcripts. From now on, let us fix an interactive *public-coin* proof $\mathsf{IP} = (\mathsf{IP.Setup}, \mathsf{IP.P}, \mathsf{IP.V})$ for a relation R. Assume further this protocol has exactly r rounds of challenges. Then, we represent a (potential) *single-execution* transcript generated by an algebraic prover in different forms, depending on whether we include the representations of group elements or not. Specifically, we let the (plain) transcript be $\tau = (\mathsf{pp}, x, a_1, c_1, a_2, c_2, \ldots, a_r, c_r, a_{r+1})$, where pp are the generated parameters, x is the input produced by $\mathcal{P}_{\mathsf{alg}}$, $c_i \in \mathsf{Ch}_i$ for all $i \in \{1,\ldots,r\}$ are the challenges, and $a_1,\ldots,a_{r+1}$ are the prover's messages. The corresponding *extended transcript* with representations is denoted as $[\tau] = (\mathsf{pp}, [x], [a_1], c_1, [a_2], c_2, \ldots, [a_r], c_r, [a_{r+1}])$.

In particular, the representation of each group element contained in a_i is with respect to all elements contained in $\mathsf{pp}, x, a_1, \ldots, a_{i-1}$. We let $\mathcal{T}^{\mathsf{IP}}$ be the set of all possible extended transcripts $[\tau]$. We also let $\mathcal{T}^{\mathsf{IP}}_{\mathsf{Acc}} \subseteq \mathcal{T}^{\mathsf{IP}}$ be the set of *accepting* transcripts $[\tau]$, i.e., $\mathsf{IP.V}(\tau) = 1$.

Path Extraction. We now would like to define a function e which extracts a witness from any accepting transcript $[\tau] \in \mathcal{T}^{\mathsf{IP}}_{\mathsf{Acc}}$. For a particular function e we now define the set of extended transcripts on which it succeeds in extracting a valid witness, i.e.,

$$\mathcal{T}^{\mathsf{IP},\mathsf{e},R}_{\mathsf{correct}} = \left\{[\tau] = (\mathsf{pp}, [x], \ldots) \in \mathcal{T}^{\mathsf{IP}}_{\mathsf{Acc}} \,:\, w \leftarrow \mathsf{e}([\tau]),\ (\mathsf{pp}, x, w) \in R\right\} .$$

Therefore, a natural extractor $\mathcal{E}$ just answers challenges honestly, and applies e to a path in the execution tree which defines an accepting transcript, and returns the corresponding witness w. The probability of this extractor failing can be upper bounded naïvely by the probability that the prover generates, in its execution tree, a path corresponding to an extended transcript $[\tau] \in \mathcal{T}^{\mathsf{IP}}_{\mathsf{Acc}} \setminus \mathcal{T}^{\mathsf{IP},\mathsf{e},R}_{\mathsf{correct}}$. This is however not directly helpful, as the main challenge is to actually estimate this probability.

BAD CHALLENGES. In all of our examples, the analysis of the probability of generating a transcript in $\mathcal{T}^{\mathsf{IP}}_{\mathsf{Acc}} \setminus \mathcal{T}^{\mathsf{IP},\mathsf{e},R}_{\mathsf{correct}}$ will generally consist of an *information-theoretic* and a *computational part*.

The information-theoretic part will account to choosing some *bad challenges*. We capture such choices of bad challenges by defining, for any partial extended transcript $[\tau'] = (\mathsf{pp}, [x], [a_1], c_1, \ldots, [a_i])$,a set $\mathsf{BadCh}(\tau') \subseteq \mathsf{Ch}_i$ of such bad challenges. (Crucially, whether a challenge is bad or not only depends on the extended transcript so far.) We now denote as $\mathcal{T}^{\mathsf{IP}}_{\mathsf{BadCh}}$ the set of all extended transcripts which contain at least one bad challenge. It turns out that the probability of generating such a bad challenge is easily bounded by $q \cdot \varepsilon$ for a prover making q oracle queries, assuming $|\mathsf{BadCh}(\tau')| / |\mathsf{Ch}_i| \leqslant \varepsilon$.

The only case that the extractor can now fail is if the execution tree contains an extended transcript $[\tau]$ in the set $\mathcal{T}^{\mathsf{IP},\mathsf{e},R}_{\mathsf{fail}} = \mathcal{T}^{\mathsf{IP}}_{\mathsf{Acc}} \setminus (\mathcal{T}^{\mathsf{IP},\mathsf{e},R}_{\mathsf{correct}} \cup \mathcal{T}^{\mathsf{IP}}_{\mathsf{BadCh}})$. We denote the probability that this happens in $\mathsf{SRS}^{\mathcal{P}_{\mathsf{alg}}}_{\mathsf{IP}}(\lambda)$ as $p_{\mathsf{fail}}(\mathsf{IP}, \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda)$. Generally, in all of our applications, upper bounding this probability for a suitably defined extractor will constitute the computational core of the proof – i.e., we will prove (generally tight) reductions to breaking some underlying assumption.

THE MASTER THEOREM. We are now ready to state our master theorem, which assumes the formal set up.

Theorem 1 (Master Theorem). *Let* IP *be an* $r = r(\lambda)$*-challenge public coin interactive proof for a relation* R*. Assume there exist functions* BadCh *and* e *for* IP *as described above, and let* p_{fail} *be as defined above. Let* τ' *be a partial transcript such that the challenge that comes right after is sampled from* Ch_i*. Assume that for all* $i \in \{1, \ldots, r\}$*, we have* $|\mathsf{BadCh}(\tau')| / |\mathsf{Ch}_i| \leqslant \varepsilon$*, for some* $\varepsilon \in [0, 1]$*. Then, there exists an extractor* $\mathcal{E}$ *that uses* e *such that for any non-uniform algebraic prover* $\mathcal{P}_{\mathsf{alg}}$ *making at most* $q = q(\lambda)$ *queries to its oracle, and any (computationally unbounded) distinguisher* $\mathcal{D}$*, for all* $\lambda \in \mathbb{N}^+$*,*

$$\mathsf{Adv}^{\mathsf{sr\text{-}wee}}_{\mathsf{IP},R}(\mathcal{P}_{\mathsf{alg}}, \mathcal{D}, \mathcal{E}, \lambda) \leqslant q\varepsilon + p_{\mathsf{fail}}(\mathsf{IP}, \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda) \; .$$

The time complexity of the extractor $\mathcal{E}$ *is* $O(q \cdot t_V + t_{\mathsf{e}})$ *where* t_V *is the time required to run* $\mathsf{IP.V}$ *and* t_{e} *is the time required to run* e*.*

The proof of this theorem is straightforward has been deferred to the full version [34].

4.2 The Fiat-Shamir Transform

The Fiat-Shamir transform uses a family of hash functions $\mathcal{H}$ to convert a r-challenge public coin interactive protocol (proof or argument) IP to a non-interactive argument $\mathsf{FS}[\mathsf{IP}, \mathcal{H}]$. When $\mathcal{H}$ is modelled as a random oracle, we denote the non-interactive argument using $\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}]$. In $\mathsf{FS}[\mathsf{IP}, \mathcal{H}]$, a hash function H is first sampled from $\mathcal{H}$. A proof on public parameters pp and input x is $\pi = (a_1, c_1, a_2, c_2, \ldots, a_r, c_r, a_{r+1})$, such that $c_i = H(\mathsf{pp}, x, a_1, c_1, \ldots, a_{i-1}, c_{i-1}, a_i)[:\mathsf{cLen}_i]$ for $i \in \{1, \ldots, r\}$, and $\mathsf{IP.V}$ returns 1 on input (pp, x, π).

Game $\mathsf{FS\text{-}EXT\text{-}1}^{\mathcal{P}_{\mathsf{alg}},\mathcal{E}}_{\mathsf{IP},R}(\lambda)$:

$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{IP.Setup}(1^\lambda)$; $([x], \mathsf{st}_{\mathcal{P}}) \leftarrow\!\!\$\ \mathcal{P}_{\mathsf{alg},\lambda}(\mathsf{pp})$; $H \leftarrow\!\!\$\ \Omega_{\mathsf{hLen}(\lambda)}$
$[\pi] \leftarrow\!\!\$\ \mathcal{P}^{H}_{\mathsf{alg},\lambda}(\mathsf{st}_{\mathcal{P}})$; $(a_1, c_1, \ldots, a_r, c_r, a_{r+1}) \leftarrow \pi$
$\mathsf{accept} \leftarrow (\mathsf{IP.V}(\mathsf{pp}, x, \pi) = 1) \wedge (\forall i \in [r] : c_i = H(\mathsf{pp}, x, a_1, c_1, \ldots, a_i)[:\mathsf{cLen}_i])$
$w \leftarrow\!\!\$\ \mathcal{E}(1^\lambda, \mathsf{pp}, [x], [\pi])$; Return $(\mathsf{accept} \wedge (\mathsf{pp}, x, w) \notin R)$

Fig. 4. Definition of fs-ext-1 security in the AGM. The game FS-EXT-1 defines fs-ext-1 security in the AGM for a non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$, an extractor $\mathcal{E}$ and a non-interactive argument obtained by applying the Fiat-Shamir transform to an interactive protocol IP. Here, IP has $r = r(\lambda)$ challenges where the i^{th} challenge is of length $\mathsf{cLen}_i = \mathsf{cLen}_i(\lambda)$ such that $\mathsf{sLen}(\lambda) \leqslant \mathsf{cLen}_i(\lambda) \leqslant \mathsf{hLen}(\lambda)$. The set $\Omega_{\mathsf{hLen}(\lambda)}$ contains all functions mapping $\{0,1\}^*$ to $\{0,1\}^{\mathsf{hLen}(\lambda)}$.

FS-EXT-1 SECURITY. We formalize a notion of proof-of-knowledge (PoK) security in the AGM for non-interactive arguments obtained by applying the Fiat-Shamir transform to an interactive protocol IP. For simplicity, this notion just captures extractability instead of witness-extended emulation. We define a notion of soundness called fs-ext-1 that captures the setting where the prover has to commit to the instance beforehand. It is formally defined using the game FS-EXT-1 in Figure 4.

For an interactive proof IP and an associated relation R, algebraic prover $\mathcal{P}_{\mathsf{alg}}$, and an extractor $\mathcal{E}$, we define $\mathsf{Adv}^{\mathsf{fs\text{-}ext\text{-}1}}_{\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}],R}(\mathcal{P}_{\mathsf{alg}}, \mathcal{E}, \lambda) = \Pr\left[\mathsf{FS\text{-}EXT\text{-}1}^{\mathcal{P}_{\mathsf{alg}},\mathcal{E}}_{\mathsf{IP},R}(\lambda)\right]$.

The following theorem connects the online srs-wee security of a public-coin protocol IP and the fs-ext-1 soundness of non-interactive protocol $\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}]$, obtained by applying the Fiat-Shamir transform using a random oracle.

Theorem 2. *Let R be a relation. Let IP be a $r = r(\lambda)$-challenge public coin interactive protocol for the relation R where the length of the i^{th} challenge is $\mathsf{cLen}_i(\lambda)$ such that $\mathsf{sLen}(\lambda) \leqslant \mathsf{cLen}_i(\lambda) \leqslant \mathsf{hLen}(\lambda)$ for $i \in \{1, \ldots, r\}$. Let $\mathcal{E}$ be an extractor for IP. We can construct an extractor $\mathcal{E}^*$ for $\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}]$ such that for every non-uniform algebraic prover $\mathcal{P}^*_{\mathsf{alg}}$ against $\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}]$ that makes $q = q(\lambda)$ random oracle queries, there exists a non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$ and $\mathcal{D}$ such that for all $\lambda \in \mathbb{N}^+$,*

$$\mathsf{Adv}^{\mathsf{fs\text{-}ext\text{-}1}}_{\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}],R}(\mathcal{P}^*_{\mathsf{alg}}, \mathcal{E}^*, \lambda) \leqslant \mathsf{Adv}^{\mathsf{sr\text{-}wee}}_{\mathsf{IP},R}(\mathcal{P}_{\mathsf{alg}}, \mathcal{D}, \mathcal{E}, \lambda) + (q+1)/2^{\mathsf{sLen}(\lambda)} \ .$$

Moreover, $\mathcal{P}_{\mathsf{alg}}$ makes at most q queries to its oracle and is nearly as efficient as $\mathcal{P}^*_{\mathsf{alg}}$. The extractor $\mathcal{E}^*$ is nearly as efficient as $\mathcal{E}$.

This proof of this theorem is deferred to the full version [34].

In the above theorem we considered challenges in IP to be bitstrings – however, this can be adapted to protocols where the challenges are from sets that are not bitstrings. The denominator of the fraction of the bound would become the size of smallest set from which the challenges are sampled, e.g., if the challenges in the a protocol were all from the set $\mathbb{Z}^*_p$, the fraction would become $(q+1)/(p-1)$.

Game $\mathsf{FS\text{-}EXT\text{-}2}^{\mathcal{P}_{\mathsf{alg}},\mathcal{E}}_{\mathsf{IP},R}(\lambda)$:

$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{IP.Setup}(1^\lambda)$; $H \leftarrow\!\!\$\ \Omega_{\mathsf{hLen}(\lambda)}$; $([x],[\pi]) \leftarrow\!\!\$\ \mathcal{P}^H_{\mathsf{alg},\lambda}(\mathsf{pp})$
$(a_1, c_1, \ldots, a_r, c_r, a_{r+1}) \leftarrow \pi$
$\mathsf{accept} \leftarrow (\mathsf{IP.V}(\mathsf{pp}, x, \pi) = 1) \wedge (\forall i \in [r] : c_i = H(\mathsf{pp}, x, a_1, c_1, \ldots, a_i)[:\mathsf{cLen}_i])$
$w \leftarrow\!\!\$\ \mathcal{E}(1^\lambda, \mathsf{pp}, [x], [\pi])$; Return $(\mathsf{accept} \wedge (\mathsf{pp}, x, w) \notin R)$

Fig. 5. Definition of fs-ext-2 security in the AGM. The game FS-EXT-2 defines fs-ext-2 security in the AGM for a non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$, an extractor $\mathcal{E}$ and a non-interactive argument obtained by applying the Fiat-Shamir transform to an interactive protocol IP. Here, IP has $r = r(\lambda)$ challenges where the i^{th} challenge is of length $\mathsf{cLen}_i = \mathsf{cLen}_i(\lambda)$ such that $\mathsf{sLen}(\lambda) \leqslant \mathsf{cLen}_i(\lambda) \leqslant \mathsf{hLen}(\lambda)$. The set $\Omega_{\mathsf{hLen}(\lambda)}$ contains all functions mapping $\{0,1\}^*$ to $\{0,1\}^{\mathsf{hLen}(\lambda)}$.

We can also consider an adaptive notion of soundness where the prover can output the instance and proof together – we call this notion fs-ext-2. It is formally defined using the game FS-EXT-2 in Figure 5. Unlike fs-ext-1, here the prover need not commit to the instance beforehand and can output the instance and proof together. For an interactive proof IP and an associated relation R, algebraic prover $\mathcal{P}_{\mathsf{alg}}$, and an extractor $\mathcal{E}$, we define $\mathsf{Adv}^{\mathsf{fs\text{-}ext\text{-}2}}_{\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}],R}(\mathcal{P}_{\mathsf{alg}}, \mathcal{E}, \lambda) = \mathsf{Pr}\left[\mathsf{FS\text{-}EXT\text{-}2}^{\mathcal{P}_{\mathsf{alg}},\mathcal{E}}_{\mathsf{IP},R}(\lambda)\right]$.

We assume that IP has BadCh, e functions as described previously. Further, we assume $\mathcal{T}^{\mathsf{IP}}_{\mathsf{BadCh}}$ is defined as above. We use $p_{\mathsf{fail},\mathsf{FS}}(\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}], \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda)$ to denote the probability that in $\mathsf{FS\text{-}EXT\text{-}2}^{\mathcal{P}_{\mathsf{alg}},\mathcal{E}}_{\mathsf{IP},R}$, $\mathcal{P}_{\mathsf{alg}}$ outputs $([x],[\pi])$, accept is $\mathtt{true}$, $\pi \notin \mathcal{T}^{\mathsf{IP}}_{\mathsf{BadCh}}$ but e on input $([x],[\pi])$ fails to produce a valid witness. The following theorem upper bounds the fs-ext-2 soundness of non-interactive protocol $\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}]$.

Theorem 3. *Let* IP *be an $r = r(\lambda)$-challenge public coin interactive proof for a relation R where the length of the i^{th} challenge is $\mathsf{cLen}_i(\lambda)$ such that $\mathsf{sLen}(\lambda) \leqslant \mathsf{cLen}_i(\lambda) \leqslant \mathsf{hLen}(\lambda)$ for $i \in \{1, \ldots, r\}$. Assume there exist functions* BadCh *and* e *as described previously and let $p_{\mathsf{fail},\mathsf{FS}}$ be as described above. Let τ' be a partial transcript such that the challenge that comes right after is sampled from Ch_i. Assume that for all $i \in \{1, \ldots, r\}$, we have that $|\mathsf{BadCh}(\tau')| \,/\, |\mathsf{Ch}_i| \leqslant \varepsilon$ for some $\varepsilon \in [0,1]$. Then, there exists an extractor $\mathcal{E}^*$ that uses* e *such that for any non-uniform algebraic prover $\mathcal{P}^*_{\mathsf{alg}}$ for $\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}]$ making at most $q = q(\lambda)$ queries to its random oracle, for all $\lambda \in \mathbb{N}^+$,*

$$\mathsf{Adv}^{\mathsf{fs\text{-}ext\text{-}2}}_{\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}],R}(\mathcal{P}^*_{\mathsf{alg}}, \mathcal{E}^*, \lambda) \leqslant q\varepsilon + p_{\mathsf{fail},\mathsf{FS}}(\mathsf{FS}^{\mathbf{RO}}[\mathsf{IP}], \mathcal{P}^*_{\mathsf{alg}}, \mathsf{e}, R, \lambda)\ .$$

The time complexity of the extractor $\mathcal{E}^*$ is $O(q \cdot t_V + t_{\mathsf{e}})$ where t_V is the time required to run IP.V *and t_{e} is the time required to run* e.

The proof of this theorem is similar to Theorem 1 and has been omitted.

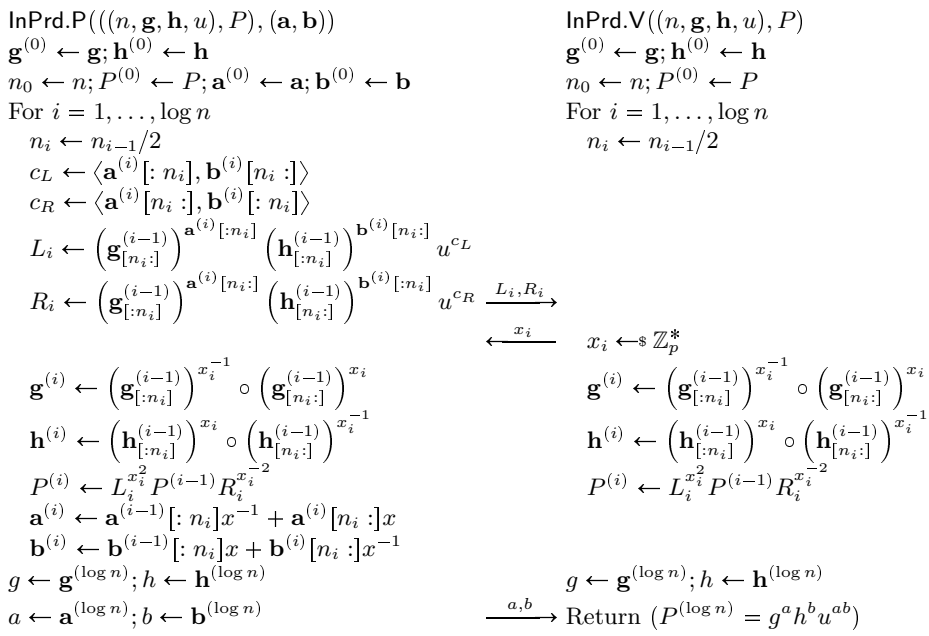

Fig. 6. Bulletproofs inner-product argument InPrd.

5 Online srs-wee Security of Bulletproofs

In this section, we shall apply our framework to prove online srs-wee security in the AGM for two instantiations of Bulletproofs- range proofs (RngPf) and proofs for arithmetic circuit satisfiability (ACSPf). We first introduce the Bulletproofs inner product argument (InPrd) in Section 5.1 which forms the core of both RngPf and ACSPf. Then, in Sections 5.2 and 5.3 we introduce and analyze online srs-wee security of RngPf and ACSPf respectively.

5.1 Inner Product Argument InPrd

We shall assume that $\mathsf{InPrd} = \mathsf{InPrd}[\mathbb{G}]$ is instantiated on an understood family of groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda \in \mathbb{N}^+}$ of order $p = p(\lambda)$. Using InPrd, a prover can convince a verifier that $P \in \mathbb{G}$ is a well-formed commitment to vectors $\mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n$ and their inner-product $\langle \mathbf{a}, \mathbf{b} \rangle$. More precisely, the prover wants to prove to the verifier that $P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} u^{\langle \mathbf{a}, \mathbf{b} \rangle}$ where $\mathbf{g} \in \mathbb{G}^n, \mathbf{h} \in \mathbb{G}^n, u \in \mathbb{G}$ are independent generators of $\mathbb{G}$. We assume that n is a power of 2 without loss of generality since if needed, one can pad the input appropriately to ensure that this holds. The prover and the verifier for InPrd is formally defined in Figure 6.

5.2 Online srs-wee Security of RngPf

We shall assume that $\mathsf{RngPf} = \mathsf{RngPf}[\mathbb{G}]$ is instantiated on an understood family of groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda \in \mathbb{N}^+}$ of order $p = p(\lambda)$. The argument RngPf is an argument

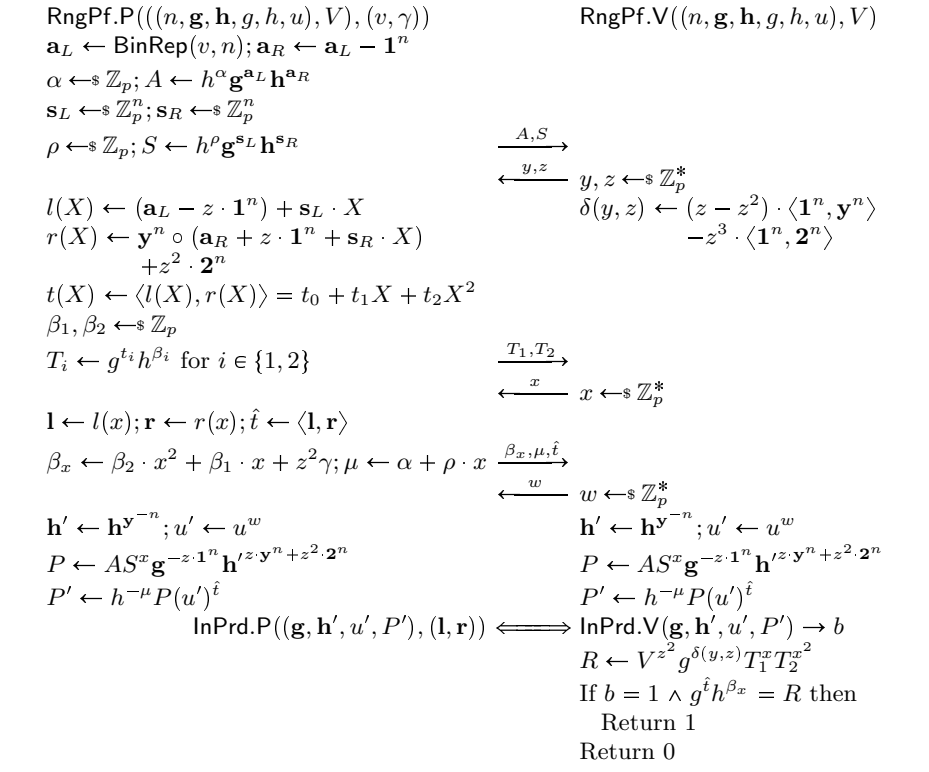

Fig. 7. Prover and Verifier for RngPf. The function $\mathsf{BinRep}(v, n)$ outputs the n-bit representation of v. The symbol $\Longleftrightarrow$ denotes the interaction between InPrd.P and InPrd.V with the output of the InPrd.V being b.

of knowledge for the relation

$$R = \left\{ \left((n \in \mathbb{N}, g, h \in \mathbb{G}), V \in \mathbb{G}, (v, \gamma \in \mathbb{Z}_p) \right) : g^v h^\gamma = V \wedge v \in [0, 2^n - 1] \right\} . \quad (4)$$

Description of RngPf. RngPf.Setup returns $\mathbf{g} \in \mathbb{G}^n, \mathbf{h} \in \mathbb{G}^n, g, h, u \in \mathbb{G}$ where $\mathbf{g}, \mathbf{h}$ are vectors of independent generators and g, h, u are other independent generators of the group $\mathbb{G}$. The prover and verifier for RngPf are defined in Figure 7.

In Theorem 4, we analyze the online srs-wee security for RngPf. Since RngPf has a group element V in its input, the analysis of non-adaptive srs-wee security would differ from the online srs-wee analysis. In the full version [34], we analyse the non-adaptive srs-wee security of RngPf – it turns out that the proof is even harder for this case because the function e does not have the representation of V. The resulting bound is increased to the square root of the adaptive bound, due to our limited use of rewinding.

Theorem 4. *Let $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ be a family of groups of order $p = p(\lambda)$. Let $\mathsf{RngPf} = \mathsf{RngPf}[\mathbb{G}]$ be the interactive argument as defined in Figure 7, for the relation R in (4). We can construct an extractor $\mathcal{E}$ such that for any non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$ making at most $q = q(\lambda)$ queries to its oracle, there exists a non-uniform adversary $\mathcal{F}$ with the property that for any (computationally unbounded) distinguisher $\mathcal{D}$, for all $\lambda \in \mathbb{N}^+$,*

$$\mathsf{Adv}^{\mathsf{sr\text{-}wee}}_{\mathsf{RngPf},R}(\mathcal{P}_{\mathsf{alg}},\mathcal{D},\mathcal{E},\lambda) \leqslant (14n+8)q/(p-1) + \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F},\lambda) + 1/p\,.$$

Moreover, the time complexity of the extractor $\mathcal{E}$ is $O(q\cdot n)$ and that of adversary $\mathcal{F}$ is $O(q\cdot n)$.

We show that the bound above is tight in Theorem 5. Using Theorem 2, we get the following corollary.

Corollary 1. *Let $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ be a family of groups of order $p = p(\lambda)$. Let $\mathsf{RngPf} = \mathsf{RngPf}[\mathbb{G}]$ be the interactive argument as defined in Figure 7, for the relation R in (4). Let $\mathsf{FS}^{\mathbf{RO}}[\mathsf{RngPf}]$ be the non-interactive argument obtained by applying the Fiat-Shamir transform to RngPf using a random oracle. We can construct an extractor $\mathcal{E}$ such that for any non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$ making at most $q = q(\lambda)$ queries to the random oracle there exists a non-uniform adversary $\mathcal{F}$ with the property that for all $\lambda \in \mathbb{N}^+$,*

$$\mathsf{Adv}^{\mathsf{fs\text{-}ext\text{-}1}}_{\mathsf{FS}^{\mathbf{RO}}[\mathsf{RngPf}],R}(\mathcal{P}_{\mathsf{alg}},\mathcal{E},\lambda) \leqslant ((14n+9)q+1)/(p-1) + \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F},\lambda) + 1/p\,.$$

Moreover, the time complexity of the extractor $\mathcal{E}$ is $O(q\cdot n)$ and that of adversary $\mathcal{F}$ is $O(q\cdot n)$.

In order to prove Theorem 4, we invoke Theorem 1 by defining BadCh and e and showing that $\varepsilon \leqslant (14n+8)/(p-1)$ and there exists an adversary $\mathcal{F}$ such that $p_{\mathsf{fail}}(\mathsf{RngPf},\mathcal{P}_{\mathsf{alg}},\mathsf{e},R,\lambda) \leqslant \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F}) + 1/p$. In more detail, we construct a function h such that for an accepting transcript $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$ if $\mathsf{e}([\tau])$ fails to produce a valid witness, then $\mathsf{h}([\tau])$ returns a non-trivial discrete logarithm relation with respect to the generators. This h is used to construct an adversary $\mathcal{H}$ against the discrete logarithm relation problem and we invoke Lemma 2 to transform into adversary $\mathcal{F}$ against the discrete logarithm problem, thus upper bounding $p_{\mathsf{fail}}(\mathsf{RngPf},\mathcal{P}_{\mathsf{alg}},\mathsf{e},R,\lambda)$ using $\mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F})$.

Proof (Theorem 4). We extend the notation for representation of group elements introduced in Section 4 for representation with respect to vector of group elements like $\mathbf{g}$. The representation of a group element $A = \mathbf{g}^{a_{\mathbf{g}}} g^{a_g}$ with respect to $(\mathbf{g}, g)$ is $[A] = (A, a_{\mathbf{g}}, a_g)$ where $a_{\mathbf{g}} = (a_{g_1}, \cdots, a_{g_n})$.

Defining BadCh and upper bounding ε. To start off, we define $\mathsf{BadCh}(\tau')$ for all partial transcripts τ'. Let Ch be the set from which the challenge that just follows τ' is sampled. We use a helper function $\mathsf{CheckBad}$ to define $\mathsf{BadCh}(\tau')$. The function $\mathsf{CheckBad}$ takes as input a partial extended transcript $[\tau']$ and a challenge $c \in \mathsf{Ch}$ and returns `true` if and only if $c \in \mathsf{BadCh}(\tau')$. For each verifier challenge in RngPf, there is a definition of $\mathsf{CheckBad}$ in Figure 8. Every $\mathsf{CheckBad}$

function defines several bad conditions that depend on τ' – most of these bad conditions are checked using the predicate SZ. This predicate takes as input a vector of polynomials and a corresponding vector of points to evaluate the polynomial on and returns `true` iff any of the polynomials is non-zero but its evaluation at the corresponding point is zero. One can safely ignore the details of the definitions of $\mathsf{CheckBad}$ functions for now – the rationale behind their definitions shall become apparent later on.

The following lemma establishes an upper bound of $(14n+8)/(p-1)$ on $|\mathsf{BadCh}(\tau')|/|\mathsf{Ch}|$.

Lemma 3. *Let τ' be a partial transcript for RngPf. Let Ch be the set from which the challenge that comes right after τ' is sampled. Then, $|\mathsf{BadCh}(\tau')|/|\mathsf{Ch}| \leqslant (14n+8)/(p-1)$.*

The proof of this lemma has been deferred to the full version [34].

Defining e. Let τ be a transcript of RngPf as defined below.

$$\tau = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), V; (A, S), (y, z), (T_1, T_2), x, (\beta_x, \mu, \hat{t}), w, (L_1, R_1), x_1, \\ (L_2, R_2), x_2, \ldots, (L_{\log n}, R_{\log n}), x_{\log n}, (a, b)\big) \ . \tag{5}$$

Let us represent using $\tau|_c$ the prefix of τ just before the challenge c. For example $\tau|_{(y,z)} = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), V, (A, S)\big)$. The function e simply returns (v_g, v_h). However, its output is a valid witness only if $v_{\mathbf{g}} = v_{\mathbf{h}} = \mathbf{0}^n, v_u = 0$ and $v_g \in [0, 2^n - 1]$.

Proving an upper bound on $p_{\mathsf{fail}}(\mathsf{RngPf}, \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda)$. We construct an adversary $\mathcal{H}$ against the discrete logarithm relation problem that takes as input independent generators $\mathbf{g}, \mathbf{h}, g, h, u$ of the group $\mathbb{G}$ and works as follows. It simulates the game $\mathsf{SRS}_{\mathsf{RngPf}}$ to $\mathcal{P}_{\mathsf{alg}}$ using public parameters $n, \mathbf{g}, \mathbf{h}, g, h, u$. If $\mathcal{P}_{\mathsf{alg}}$ manages to produce an accepting transcript τ, $\mathcal{H}$ calls a helper function h on input $\lceil\tau\rceil$ and outputs whatever h outputs. We shall define h in such a way that for $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$ if $\mathsf{e}(\lceil\tau\rceil)$ does not return a valid witness, then $\mathsf{h}(\lceil\tau\rceil)$ returns a non-trivial discrete logarithm relation. In other words, we have that whenever $\mathsf{e}(\lceil\tau\rceil)$ fails to extract a valid witness for an accepting transcript $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$, $\mathcal{H}$ succeeds. So we have that $p_{\mathsf{fail}}(\mathsf{RngPf}, \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda) \leqslant \mathsf{Adv}^{\mathsf{dl\text{-}rel}}_{\mathbb{G}, 2n+3}(\mathcal{H})$. Using Lemma 2 we would have that there exists an adversary $\mathcal{F}$ such that $p_{\mathsf{fail}}(\mathsf{RngPf}, \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda) \leqslant \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F}) + 1/p$. We also have that $\mathcal{F}$ is nearly as efficient as $\mathcal{H}$.

Defining h. We next describe the h function. Let τ, as defined in (5), be an accepting transcript. $V^{z^2} g^{\delta(y,z)} T_1^x T_2^{x^2} = g^{\hat{t}} h^{\beta_x}$. must hold since τ is an accepting transcript.

The function h can plug in the representations of T_1, T_2, V into the above equation and compute $e^{(1)}_{\mathbf{g}}, e^{(1)}_{\mathbf{h}}, e^{(1)}_g, e^{(1)}_h, e^{(1)}_u$ such that $\mathbf{g}^{e^{(1)}_{\mathbf{g}}} \mathbf{h}^{e^{(1)}_{\mathbf{h}}} g^{e^{(1)}_g} h^{e^{(1)}_h} u^{e^{(1)}_u} = 1$. If not all of these are zero, h returns $e^{(1)}_{\mathbf{g}}, e^{(1)}_{\mathbf{h}}, e^{(1)}_g, e^{(1)}_h, e^{(1)}_u$.

Again since τ is an accepting transcript, $\mathsf{InPrd.V}$ must have returned 1 and hence $P^{(\log n)} = (\mathbf{g}^{(\log n)})^a (\mathbf{h}^{(\log n)})^b u^{ab}$ must hold. All the terms in the above equality can be expressed in terms of $\mathbf{g}, \mathbf{h}, g, h, u$ and one can compute

Procedure $\mathsf{CheckBad}(\lfloor\tau'\rfloor, (y,z))$:

$//\lfloor\tau'\rfloor = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), [V], ([A], [S])\big)$
$f(Y,Z) \leftarrow Z^2(v_g - \langle a_{\mathbf{g}}, \mathbf{2}^n\rangle) - Z\langle a_{\mathbf{g}} - a_{\mathbf{h}} - \mathbf{1}^n, \mathbf{Y}^n\rangle - \langle a_{\mathbf{g}} \circ a_{\mathbf{h}}, \mathbf{Y}^n\rangle$
Return $\mathsf{SZ}(f(Y,Z), (y,z))$

Procedure $\mathsf{CheckBad}(\lfloor\tau'\rfloor, x)$:

$//\lfloor\tau'\rfloor = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), [V], ([A], [S]), (y,z), ([T_1], [T_2])\big)$
$f_1(X) \leftarrow v_{\mathbf{g}} z^2 + t_{1\mathbf{g}} X + t_{2\mathbf{g}} X^2$; $f_2(X) \leftarrow v_{\mathbf{h}} z^2 + t_{1\mathbf{h}} X + t_{2\mathbf{h}} X^2$
$f_3(X) \leftarrow v_u z^2 + t_{1u} X + t_{2u} X^2$; $\delta(y,z) \leftarrow (z - z^2)\langle \mathbf{1}^n, \mathbf{y}^n\rangle - z^3\langle \mathbf{1}^n, \mathbf{2}^n\rangle$
$l(X) \leftarrow (a_{\mathbf{g}} - z \cdot \mathbf{1}^n) + s_{\mathbf{g}} \cdot X$; $r(X) \leftarrow \mathbf{y}^n \circ (a_{\mathbf{h}} + z \cdot \mathbf{1}^n + s_{\mathbf{h}} \cdot X) + z^2 \cdot \mathbf{2}^n$
$f_4(X) \leftarrow v_g z^2 + \delta(y,z) + t_{1g} X + t_{2g} X^2 - \langle l(X), r(X)\rangle$
Return $\mathsf{SZ}(f_1(X), x) \vee \mathsf{SZ}(f_2(X), x) \vee \mathsf{SZ}(f_3(X), x) \vee \mathsf{SZ}(f_4(X), x)$

Procedure $\mathsf{CheckBad}(\lfloor\tau'\rfloor, w)$:

$//\lfloor\tau'\rfloor = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), [V], ([A], [S]), (y,z), ([T_1], [T_2]), x, (\beta_x, \mu, \hat{t})\big)$
$\mathbf{l} \leftarrow (a_{\mathbf{g}} - z \cdot \mathbf{1}^n) + s_{\mathbf{g}} \cdot x$; $\mathbf{r} \leftarrow (a_{\mathbf{h}} + x s_{\mathbf{h}} + z\mathbf{1}^n) \circ \mathbf{y}^n + z^2 \mathbf{2}^n$; $f(W) \leftarrow W\hat{t} - W\langle \mathbf{l}, \mathbf{r}\rangle$
Return $\mathsf{SZ}(f(W), w)$

Procedure $\mathsf{CheckBad}(\lfloor\tau'\rfloor, x_m)$:

$//\lfloor\tau'\rfloor = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), [V], ([A], [S]), (y,z), ([T_1], [T_2]), x, (\beta_x, \mu, \hat{t}), w,$
$\quad ([L_1], [R_1]), x_1, \ldots, ([L_m], [R_m])\big)$
$p'_{\mathbf{g}} \leftarrow a_{\mathbf{g}} + x s_{\mathbf{g}} - z\mathbf{1}^n$; $p'_{\mathbf{h}} \leftarrow a_{\mathbf{h}} + x s_{\mathbf{h}} + \mathbf{y}^{-n} \circ (z\mathbf{y}^n + z^2\mathbf{2}^n)$; $p'_u \leftarrow a_u + x s_u + w\hat{t}$
For $j = 0, \ldots, n-1$ do
  $f^{\mathbf{g}}_{m,j}(X) \leftarrow l_{mg_{1+j}} + X^2 + r_{mg_{1+j}} X^{-2} + p'_{g_{1+j}} + \sum_{i=1}^{m-1} (l_{ig_{1+j}} x_i^2 + r_{ig_{1+j}} x_i^{-2})$
  $f^{\mathbf{h}}_{m,j}(X) \leftarrow l_{mh_{1+j}} X^2 + r_{mh_{1+j}} X^{-2} + p'_{h_{1+j}} + \sum_{i=1}^{m-1} (l_{ih_{1+j}} x_i^2 + r_{ih_{1+j}} x_i^{-2})$
$f^u_m(X) \leftarrow l_{mu} X^2 + r_{mu} X^{-2} + p'_u + \sum_{i=1}^{m-1} (l_{iu} x_i^2 + r_{iu} x_i^{-2})$
$\mathsf{flag} \leftarrow \mathtt{false}$
For $t = 1, \ldots, m-1$ do for $j = 0, \ldots, n/2^t - 1$ do
  $\mathsf{flag} \leftarrow \mathsf{flag} \vee \mathsf{SZ}(f^{\mathbf{g}}_{m,j}(X) \cdot x_t^2 - f^{\mathbf{g}}_{m,j+n/2^t}(X), x_m) \vee \mathsf{SZ}(f^{\mathbf{h}}_{m,j}(X) - f^{\mathbf{h}}_{m,j+n/2^t}(X) \cdot x_t^2, x_m)$
For $j = 0, \ldots, n/2^m - 1$ do
  $\mathsf{flag} \leftarrow \mathsf{flag} \vee \mathsf{SZ}(f^{\mathbf{g}}_{m,j}(X) \cdot X^2 - f^{\mathbf{g}}_{m,j+n/2^m}(X), x_m) \vee \mathsf{SZ}(f^{\mathbf{h}}_{m,j}(X) - f^{\mathbf{h}}_{m,j+n/2^m}(X) \cdot X^2, x_m)$
$\mathsf{flag} \leftarrow \mathsf{flag} \vee \mathsf{SZ}\Big(f^u_m(X) - w \cdot \sum_{j=0}^{n/2^m-1} f^{\mathbf{g}}_{m,j}(X) \cdot f^{\mathbf{h}}_{m,j}(X) \cdot y^j, x_m\Big)$
Return flag

Fig. 8. The functions $\mathsf{CheckBad}$ function for the RngPf.

Procedure $\mathsf{e}([\tau])$:

$//[\tau] = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), [V]; ([A], [S]), (y,z), ([T_1], [T_2]), x, (\beta_x, \mu, \hat{t}), w,$
$\quad ([L_1], [R_1]), x_1, \ldots, ([L_{\log n}], [R_{\log n}]), x_{\log n}, (a,b)\big)$
$v^* \leftarrow v_g$; $\gamma^* \leftarrow v_h$; Return (v^*, γ^*)

Fig. 9. The function e for RngPf.

$e^{(2)}_{\mathbf{g}}, e^{(2)}_{\mathbf{h}}, e^{(2)}_g, e^{(2)}_h, e^{(2)}_u$ such that $\mathbf{g}^{e^{(2)}_{\mathbf{g}}} \mathbf{h}^{e^{(2)}_{\mathbf{h}}} g^{e^{(2)}_g} h^{e^{(2)}_h} u^{e^{(2)}_u} = 1$. The function h computes and returns $e^{(2)}_{\mathbf{g}}, e^{(2)}_{\mathbf{h}}, e^{(2)}_g, e^{(2)}_h, e^{(2)}_u$. We define the function h formally in

Procedure $\mathsf{h}([\tau])$:

$//[\tau] = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), [V], ([A], [S]), (y, z), ([T_1], [T_2]), x, (\beta_x, \mu, \hat{t}), w,$
$([L_1], [R_1]), x_1, \ldots, ([L_{\log n}], [R_{\log n}]), x_{\log n}, (a, b)\big)$

$\delta(y,z) \leftarrow (z - z^2)\langle \mathbf{1}^n, \mathbf{y}^n \rangle - z^3 \langle \mathbf{1}^n, \mathbf{2}^n \rangle$

$e^{(1)}_{\mathbf{g}} \leftarrow v_{\mathbf{g}} z^2 + t_{1\mathbf{g}} x + t_{2\mathbf{g}} x^2$; $e^{(1)}_{\mathbf{h}} \leftarrow v_{\mathbf{h}} z^2 + t_{1\mathbf{h}} x + t_{2\mathbf{h}} x^2$; $e^{(1)}_{u} \leftarrow v_{u} z^2 + t_{1u} x + t_{2u} x^2$

$e^{(1)}_{g} \leftarrow v_{g} z^2 + \delta(y,z) + t_{1g} x + t_{2g} x^2 - \hat{t}$; $e^{(1)}_{h} \leftarrow v_{h} z^2 + t_{1h} x + t_{2h} x^2 - \beta_x$

If $(e^{(1)}_{\mathbf{g}}, e^{(1)}_{\mathbf{h}}, e^{(1)}_{u}, e^{(1)}_{g}, e^{(1)}_{h}) \neq (\mathbf{0}^n, \mathbf{0}^n, 0, 0, 0)$ then return $(e^{(1)}_{\mathbf{g}}, e^{(1)}_{\mathbf{h}}, e^{(1)}_{u}, e^{(1)}_{g}, e^{(1)}_{h})$

$p'_{\mathbf{g}} \leftarrow (a_{\mathbf{g}}) + x s_{\mathbf{g}} - z\mathbf{1}^n$; $p'_{\mathbf{h}} \leftarrow a_{\mathbf{h}} + x s_{\mathbf{h}} + \mathbf{y}^{-n} \circ (z\mathbf{y}^n + z^2 \mathbf{2}^n)$

$p'_{g} \leftarrow a_g + x s_g$; $p'_{h} \leftarrow a_h + x s_h - \mu$; $p'_{u} \leftarrow a_u + x s_u + w\hat{t}$

For $k = 0$ to $n-1$ do

$e^{(2)}_{g_{k+1}} \leftarrow p'_{g_{1+k}} + \sum_{i=1}^{\log n} l_{ig_{1+k}} x_i^2 + r_{ig_{1+k}} x_i^{-2} - a \cdot \prod_{i=1}^{\log n} x_i^{(-1)^{1-\mathsf{bit}(k,i,\log n)}}$

$e^{(2)}_{h_{k+1}} \leftarrow p'_{h_{1+k}} + \sum_{i=1}^{\log n} l_{ih_{1+k}} x_i^2 + r_{ih_{1+k}} + x_i^{-2} - b y^{(-(k))} \cdot \prod_{i=1}^{\log n} x_i^{(-1)^{\mathsf{bit}(k,i,\log n)}}$

$e^{(2)}_{\mathbf{g}} \leftarrow (e^{(2)}_{g_1}, \ldots, e^{(2)}_{g_n})$; $e^{(2)}_{\mathbf{h}} \leftarrow (e^{(2)}_{h_1}, \ldots, e^{(2)}_{h_n})$; $e^{(2)}_{u} \leftarrow p'_u + \sum_{i=1}^{\log n} l_{iu} x_i^2 + r_{iu} x_i^{-2} - w \cdot ab$

$e^{(2)}_{g} \leftarrow \sum_{i=1}^{\log n} l_{ig} x_i^2 + r_{ig} x_i^{-2} + p'_g$; $e^{(2)}_{h} \leftarrow \sum_{i=1}^{\log n} l_{ih} x_i^2 + r_{ih} x_i^{-2} + p'_h$

Return $(e^{(2)}_{\mathbf{g}}, e^{(2)}_{\mathbf{h}}, e^{(2)}_{u}, e^{(2)}_{g}, e^{(2)}_{h})$

Fig. 10. The function h for RngPf.

Figure 10. It follows from the description of h that it runs in time $O(n)$. The running time of $\mathcal{H}$ consists of the time required to answers q queries, run $\mathsf{RngPf.V}$ in at most q paths in the execution tree and the time required to run h. Hence its time complexity is $O(q \cdot n)$. Using Lemma 2, time complexity of $\mathcal{F}$ is $O(q \cdot n)$.

RELATING h, e. In order to complete the proof of Theorem 4, in the following lemma we show that – for an accepting transcript τ such that $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$ if $\mathsf{e}([\tau])$ does not return a valid witness, then $\mathsf{h}([\tau])$ returns a non-trivial discrete logarithm relation. Proving this lemma would conclude the proof of Theorem 4.

Lemma 4. *Let τ, as defined in (5), be an accepting transcript of* RngPf *such that $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$. If $\mathsf{e}([\tau])$ returns (v^*, γ^*) such that at least one of the following hold: $g^{v^*} h^{\gamma^*} \neq V$ or $v^* \notin [0, 2^n - 1]$, then $\mathsf{h}([\tau])$ returns a non-trivial discrete logarithm relation.*

Proof (Lemma 4). For simplicity, we shall prove the contrapositive of the statement, i.e., assuming $\mathsf{h}([\tau])$ returns a trivial discrete logarithm relation, then $g^{v^*} h^{\gamma^*} = V$ and $v^* \in [0, 2^n - 1]$.

In order to prove $g^{v^*} h^{\gamma^*} = V$ and $v^* \in [0, 2^n - 1]$, it suffices to show that $v_{\mathbf{g}} = v_{\mathbf{h}} = \mathbf{0}^n$, $v_u = 0$ and $v_g \in [0, 2^n - 1]$. Let us denote using $\tau|_c$ the partial transcript that is the prefix of τ just before the challenge c. For example $\tau|_{(y,z)} = \big((n, \mathbf{g}, \mathbf{h}, u, g, h), V, (A, S)\big)$.Since we assumed that $\mathsf{h}([\tau])$ returns $(\mathbf{0}^n, \mathbf{0}^n, 0, 0, 0)$, we have that for $i = 1, 2$, $(e^{(i)}_{\mathbf{g}}, e^{(i)}_{\mathbf{h}}, e^{(i)}_{g}, e^{(i)}_{h}, e^{(i)}_{u}) = (\mathbf{0}^n, \mathbf{0}^n, 0, 0, 0)$.

Writing out the expression for $e_{\mathbf{g}}^{(1)}$ we get $v_{\mathbf{g}}z^2 + t_{1\mathbf{g}}x + t_{2\mathbf{g}}x^2 = \mathbf{0}^n$.Since $\tau \notin \mathcal{T}_{\mathsf{BadCh}}^{\mathsf{RngPf}}$, we have that $x \notin \mathsf{BadCh}(\tau|_x)$. Therefore, $\mathsf{SZ}(f_1(X), x)$ is `false` where f_1 is as defined in $\mathsf{CheckBad}(\tau', x)$. Since we have here that $f_1(x) = 0$, the polynomial $f_1(X)$ is the zero vector polynomial. Since $z \neq 0$ it follows that $v_{\mathbf{g}} = \mathbf{0}^n$. Similarly using $e_{\mathbf{h}}^{(1)} = \mathbf{0}^n$ and $e_u^{(1)} = 0$ we can show that $v_{\mathbf{h}} = \mathbf{0}^n$ and $v_u = 0$ respectively. Writing out the expression for $e_g^{(1)}$ we have $v_g z^2 + \delta(y, z) + t_{1g}x + t_{2g}x^2 - \hat{t} = 0$. Hence,

$$\hat{t} = v_g z^2 + \delta(y, z) + t_{1g}x + t_{2g}x^2 \ . \tag{6}$$

Using $e_{\mathbf{g}}^{(2)} = \mathbf{0}^n$ we get for all $k \in \{0, \ldots, n-1\}$

$$p'_{g_{1+k}} + \sum_{i=1}^{\log n} (l_{ig_{1+k}} x_i^2 + r_{ig_{1+k}} x_i^{-2}) - a \cdot \prod_{i=1}^{\log n} x_i^{(-1)^{1-\mathsf{bit}(k,i,\log n)}} = 0 \ . \tag{7}$$

Using $e_{\mathbf{h}}^{(2)} = \mathbf{0}^n$ we get for all $k \in \{0, \ldots, n-1\}$

$$p'_{h_{1+k}} + \sum_{i=1}^{\log n} (l_{ih_{1+k}} x_i^2 + r_{ih_{1+k}} x_i^{-2}) - by^{(-(k))} \cdot \prod_{i=1}^{\log n} x_i^{(-1)^{\mathsf{bit}(k,i,\log n)}} = 0 \ . \tag{8}$$

Using $e_u^{(2)} = 0$ we get that

$$p'_u + \sum_{i=1}^{\log n} (l_{iu} x_i^2 + r_{iu} x_i^{-2}) - w \cdot ab = 0 \ . \tag{9}$$

We shall next use the following lemma which essentially says that if all of $e_{\mathbf{g}}^{(2)}, e_{\mathbf{h}}^{(2)}, e_u^{(2)}, e_g^{(2)}, e_h^{(2)}$ are zero and $\tau \notin \mathcal{T}_{\mathsf{BadCh}}^{\mathsf{RngPf}}$, then $w \cdot \langle p'_{\mathbf{g}}, p'_{\mathbf{h}} \circ \mathbf{y}^n \rangle = p'_u$.

Lemma 5. *Let τ, as shown in (5), be an accepting transcript of* RngPf *such that $\tau \notin \mathcal{T}_{\mathsf{BadCh}}^{\mathsf{RngPf}}$. Let*

$$p'_{\mathbf{g}} = a_{\mathbf{g}} + x s_{\mathbf{g}} - z\mathbf{1}^n \ , p'_{\mathbf{h}} = a_{\mathbf{h}} + x s_{\mathbf{h}} + \mathbf{y}^{-n} \circ (z\mathbf{y}^n + z^2\mathbf{2}^n) \ , p'_u = a_u + x s_u + w\hat{t} \ .$$

Suppose, the for all $k \in \{0, \ldots, n-1\}$

$$\left(\sum_{i=1}^{\log n} (l_{ig_{1+k}} x_i^2 + r_{ig_{1+k}} x_i^{-2}) + p'_{g_{1+k}} \right) - a \cdot \left(\prod_{i=1}^{\log n} x_i^{(-1)^{1-\mathsf{bit}(k,i,\log n)}} \right) = 0 \ ,$$

$$\left(\sum_{i=1}^{\log n} (l_{ih_{1+k}} x_i^2 + r_{ih_{1+k}} x_i^{-2}) + p'_{h_{1+k}} \right) - by^{(-(k))} \cdot \left(\prod_{i=1}^{\log n} x_i^{(-1)^{\mathsf{bit}(k,i,\log n)}} \right) = 0 \ .$$

Also, $\left(\sum_{i=1}^{\log n} (l_{iu} x_i^2 + r_{iu} x_i^{-2}) \right) + p'_u - w \cdot ab = 0$. *Then* $w \cdot \langle p'_{\mathbf{g}}, p'_{\mathbf{h}} \circ \mathbf{y}^n \rangle = p'_u$.

The proof of this lemma is a generalization of the proof that we gave for the inner product argument for $n = 2$ in the technical overview. We defer the proof of Lemma 5 to the full version [34].

Since τ is an accepting transcript of RngPf and $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$ and (7) to (9) hold, using Lemma 5, we get $w\langle p'_{\mathbf{g}}, p'_{\mathbf{h}} \circ \mathbf{y}^n \rangle = p'_u$. Plugging in the values of $p'_{\mathbf{g}}, p'_{\mathbf{h}}, p'_u$ we get

$$w \cdot \langle a_{\mathbf{g}} + x s_{\mathbf{g}} - z\mathbf{1}^n, (a_{\mathbf{h}} + x s_{\mathbf{h}} + z\mathbf{1}^n) \circ \mathbf{y}^n + z^2 \mathbf{2}^n \rangle = a_u + x s_u + w\hat{t} \; .$$

Since $\tau \notin \mathcal{T}^{\mathsf{RngPf}}_{\mathsf{BadCh}}$, we have that $w \notin \mathsf{BadCh}(\tau|_w)$. Therefore, $\mathsf{SZ}(f(W), w)$ is $\mathtt{false}$ where f is as defined in $\mathsf{CheckBad}(\tau', w)$. Since we have here that $f(w) = 0$, the polynomial $f(W)$ must be the zero polynomial. In particular its W term must be zero, i.e., $\langle a_{\mathbf{g}} + x s_{\mathbf{g}} - z\mathbf{1}^n, (a_{\mathbf{h}} + x s_{\mathbf{h}} + z\mathbf{1}^n) \circ \mathbf{y}^n + z^2 \mathbf{2}^n \rangle = \hat{t}$. Plugging in the value of $\hat{t}$ obtained in (6) and using $x \notin \mathsf{BadCh}(\tau|_x)$, we have that

$$v_g z^2 + \delta(y, z) - \langle a_{\mathbf{g}} - z\mathbf{1}^n, (a_{\mathbf{h}} + z\mathbf{1}^n) \circ \mathbf{y}^n + z^2 \mathbf{2}^n \rangle = 0 \; .$$

Plugging in the value of $\delta(y, z)$, rearranging and simplifying we get

$$z^2(v_g - \langle a_{\mathbf{g}}, \mathbf{2}^n \rangle) - z\langle a_{\mathbf{g}} - a_{\mathbf{h}} - \mathbf{1}^n, \mathbf{y}^n \rangle - \langle a_{\mathbf{g}} \circ a_{\mathbf{h}}, \mathbf{y}^n \rangle = 0 \; .$$

Using $(y, z) \notin \mathsf{BadCh}(\tau|_{(y,z)})$, we get that $v_g - \langle a_{\mathbf{g}}, \mathbf{2}^n \rangle = 0$, $a_{\mathbf{g}} - a_{\mathbf{h}} - \mathbf{1}^n = \mathbf{0}^n$, $a_{\mathbf{g}} \circ a_{\mathbf{h}} = \mathbf{0}^n$. Note that $a_{\mathbf{g}} - a_{\mathbf{h}} - \mathbf{1}^n = \mathbf{0}^n$ and $a_{\mathbf{g}} \circ a_{\mathbf{h}} = \mathbf{0}^n$ imply that $a_{\mathbf{g}} \in \{0, 1\}^n$. Further $v_g - \langle a_{\mathbf{g}}, \mathbf{2}^n \rangle = 0$, i.e., $v_g = \langle a_{\mathbf{g}}, \mathbf{2}^n \rangle$. So, $v_g \in [0, 2^n - 1]$. Therefore, v^*, γ^* output by $\mathsf{e}([\tau])$ satisfy $V = g^{v^*} h^{\gamma^*}$ and $v^* \in [0, 2^n - 1]$. This concludes the proof of Lemma 4 and Theorem 4.

□

Further for a prover $\mathcal{P}_{\mathsf{alg}}$ for $\mathsf{FS}^{\mathbf{RO}}[\mathsf{RngPf}]$, and the e we define in the proof of Theorem 4, we can upper bound $p_{\mathsf{fail},\mathsf{FS}}(\mathsf{FS}^{\mathbf{RO}}[\mathsf{RngPf}], \mathcal{P}_{\mathsf{alg}}, \mathsf{e}, R, \lambda)$ using techniques very similar to those used in the proof of Theorem 4. This is because we can prove that if the prover outputs an instance and an accepting proof and e fails to produce a valid witness, then we can compute a non-trivial discrete logarithm relation from the representation of the transcript and instance unless one of the challenges in the transcript are bad which we can show happens with small probability. Then using Theorem 3 we obtain a bound for the fs-ext-2 security of $\mathsf{FS}^{\mathbf{RO}}[\mathsf{RngPf}]$ similar to the one we obtained for fs-ext-1 security in Corollary 1.

Tightness of Theorem 4. We next argue that the factor $O(nq/(p-1))$ in Theorem 4 is tight. We first note that the protocol RngPf can be used for the following relation

$$R' = \left\{ (n \in \mathbb{N}, g, V \in \mathbb{G}, v \in \mathbb{Z}_p) : g^v = V \wedge v \in [0, 2^n - 1] \right\} , \tag{10}$$

by fixing γ to 0.

We shall construct a cheating prover $\mathcal{P}$ (that makes $O(q)$ queries to $\mathbf{O}_{\text{ext}}$) for the relation R' that outputs an instance $V = g^v$ such that $v \notin [0, 2^n - 1]$ but

can still convince the RngPf verifier with probability $\Omega(nq/(p-1))$ if n divides $p-1$. This would imply that the bound in Theorem 4 is tight up to constant factors.

Theorem 5. *Let $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ be a family of groups of prime order $p = p(\lambda)$. Let $\mathsf{RngPf} = \mathsf{RngPf}[\mathbb{G}]$ be the interactive argument for the relation R' in (10) obtained by setting $\gamma = 0$ in the protocol defined in Figure 7. If n divides $p-1$, we can construct a non-uniform prover $\mathcal{P}$ making at most $q + \log n + 1$ queries to its oracle, such that for all $\lambda \in \mathbb{N}^+$, $\mathsf{Adv}^{\mathsf{srs}}_{\mathsf{RngPf}}(\mathcal{P}, \lambda) = (n-1)q/(p-1)$.*

The proof of this theorem has been deferred to the full version [34].

5.3 Online srs-wee Security for ACSPf

In this section, we introduce ACSPf and apply our framework to prove online srs-wee security. As shown in [10], any arithmetic circuit with n multiplication gates can be represented using a constraint system that has three vectors $\mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O \in \mathbb{Z}_p^n$ representing the left inputs, right inputs, and outputs of multiplication gates respectively, so that $\mathbf{a}_L \circ \mathbf{a}_R = \mathbf{a}_O$, with additional $Q \leqslant 2n$ linear constraints. The linear constraints can be represented as $\mathbf{a}_L \cdot \mathbf{W}_L + \mathbf{a}_R \cdot \mathbf{W}_R + \mathbf{a}_O \cdot \mathbf{W}_O = \mathbf{c}$, where $\mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O \in \mathbb{Z}_p^{Q\times n}$.

We shall assume that $\mathsf{ACSPf} = \mathsf{ACSPf}[\mathbb{G}]$ is instantiated on an understood family of groups $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ of order $p = p(\lambda)$. The argument ACSPf is an argument of knowledge for the relation

$$R = \Big\{ \big((n, Q \in \mathbb{N}), (\mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O \in \mathbb{Z}_p^{Q\times n}, \mathbf{c} \in \mathbb{Z}_p^Q), (\mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O \in \mathbb{Z}_p^n)\big) : \mathbf{a}_L \circ \mathbf{a}_R = \mathbf{a}_O \wedge \mathbf{W}_L \cdot \mathbf{a}_L + \mathbf{W}_R \cdot \mathbf{a}_R + \mathbf{W}_O \cdot \mathbf{a}_O = \mathbf{c} \Big\} . \tag{11}$$

The description of ACSPf is deferred to the full version [34]. We prove the following theorem that gives an upper bound on the advantage against online srs-wee security of ACSPf.

Theorem 6. *Let $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ be a family of groups of order $p = p(\lambda)$. Let $\mathsf{ACSPf} = \mathsf{ACSPf}[\mathbb{G}]$ be the Bulletproofs interactive argument system for arithmetic circuit satisfiability for the relation R in (11). We can construct an extractor $\mathcal{E}$ such that for any non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$ making at most $q = q(\lambda)$ queries to its oracle, there exists a non-uniform adversary $\mathcal{F}$ with the property that for any (computationally unbounded) distinguisher $\mathcal{D}$, for all $\lambda \in \mathbb{N}^+$, $\mathsf{Adv}^{\mathsf{sr\text{-}wee}}_{\mathsf{ACSPf},R}(\mathcal{P}_{\mathsf{alg}}, \mathcal{D}, \mathcal{E}, \lambda) \leqslant ((14n+8)q)/p - 1 + \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F}, \lambda) + 1/p$.*

Moreover, the time complexity of the extractor $\mathcal{E}$ is $O(q \cdot n)$ and that of adversary $\mathcal{F}$ is $O(q \cdot n)$.

We can show that the bound in Theorem 6 is tight by constructing a cheating prover like we did in Theorem 5. Using Theorem 2, we can get a corollary about fs-ext-1 security of $\mathsf{FS}^{\mathbf{RO}}[\mathsf{ACSPf}]$ which we include in the full version [34]. Additionally, using techniques similar to those in the proof of Theorem 6, we can prove a similar bound for fs-ext-2 security of $\mathsf{FS}^{\mathbf{RO}}[\mathsf{ACSPf}]$. The proof of Theorem 6 is similar to the proof of Theorem 4 and has been deferred to the full version [34].

6 Online srs-wee Security of Sonic

We apply our framework to prove srs-wee security of Sonic [18] which is an interactive argument for arithmetic circuit satisfiability based on pairings (we refer to this argument as $\mathsf{SnACSPf}$). The argument $\mathsf{SnACSPf}$ is again an argument of knowledge for the relation (11). The description of $\mathsf{SnACSPf}$ has been deferred to the full version [34]. We prove the following theorem that establishes an upper bound on the advantage against online srs-wee security of $\mathsf{SnACSPf}$.

Theorem 7. *Let $\mathbb{G} = \{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}^+}$ be a family of groups with order $p = p(\lambda)$. Let $\mathbb{G}_T = \{\mathbb{G}_{T,\lambda}\}_{\lambda\in\mathbb{N}^+}$ be a family of groups such that there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $\mathsf{SnACSPf} = \mathsf{SnACSPf}[\mathbb{G}, \mathbb{G}_T, e]$ be the Sonic interactive argument system for the relation R in (11). We can construct an extractor $\mathcal{E}$ such that for any non-uniform algebraic prover $\mathcal{P}_{\mathsf{alg}}$ making at most $q = q(\lambda)$ queries to its oracle, there exist non-uniform adversaries $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$ with the property that for any (computationally unbounded) distinguisher $\mathcal{D}$, for all $\lambda \in \mathbb{N}^+$,*

$$\mathsf{Adv}^{\mathsf{sr\text{-}wee}}_{\mathsf{SnACSPf},R}(\mathcal{P}_{\mathsf{alg}}, \mathcal{D}, \mathcal{E}, \lambda) \leqslant \frac{18nq}{p-1} + \mathsf{Adv}^{4n\text{-}\mathsf{dl}}_{\mathbb{G}}(\mathcal{F}_1, \lambda) + \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F}_2, \lambda) + \mathsf{Adv}^{\mathsf{dl}}_{\mathbb{G}}(\mathcal{F}_3, \lambda) \,.$$

Moreover, the time complexities of the extractor $\mathcal{E}$ and adversaries $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$ are all $O(q \cdot n)$.

We can show that the bound in Theorem 7 is tight by constructing a cheating prover like we did in Theorem 5. Using Theorem 2, we can get a corollary about fs-ext-1 security of $\mathsf{FS}^{\mathbf{RO}}[\mathsf{SnACSPf}]$ which we state in the full version [34]. Additionally, using techniques similar to those in the proof of Theorem 7, we can prove a similar bound for fs-ext-2 security of $\mathsf{FS}^{\mathbf{RO}}[\mathsf{SnACSPf}]$. The proof of Theorem 7 has been deferred to the full version [34].

Acknowledgements. We thank Joseph Jaeger for extensive discussions and his involvement in the earlier stages of this work. We thank the anonymous reviewers for helpful comments. This work was partially supported by NSF grants CNS-1930117 (CAREER), CNS-1926324, CNS-2026774, a Sloan Research Fellowship, and a JP Morgan Faculty Award.

References

1. S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems (extended abstract)," in *17th ACM STOC*, pp. 291–304, ACM Press, May 1985.
2. M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *ACM CCS 93* (D. E. Denning, R. Pyle, R. Ganesan, R. S. Sandhu, and V. Ashby, eds.), pp. 62–73, ACM Press, Nov. 1993.
3. V. Shoup, "Lower bounds for discrete logarithms and related problems," in *EUROCRYPT'97* (W. Fumy, ed.), vol. 1233 of *LNCS*, pp. 256–266, Springer, Heidelberg, May 1997.

4. U. M. Maurer, "Abstract models of computation in cryptography (invited paper)," in *10th IMA International Conference on Cryptography and Coding* (N. P. Smart, ed.), vol. 3796 of *LNCS*, pp. 1–12, Springer, Heidelberg, Dec. 2005.
5. G. Fuchsbauer, E. Kiltz, and J. Loss, "The algebraic group model and its applications," in *CRYPTO 2018, Part II* (H. Shacham and A. Boldyreva, eds.), vol. 10992 of *LNCS*, pp. 33–62, Springer, Heidelberg, Aug. 2018.
6. B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more," in *2018 IEEE Symposium on Security and Privacy*, pp. 315–334, IEEE Computer Society Press, May 2018.
7. "Monero to become first billion-dollar crypto to implement 'bulletproofs' tech." https://www.coindesk.com/monero-to-become-first-billion-dollar-crypto-to-implement-bulletproofs-tech.
8. "Signal adds a payments feature—with a privacy-focused cryptocurrency." https://www.wired.com/story/signal-mobilecoin-payments-messaging-cryptocurrency/.
9. J. Nick, T. Ruffing, Y. Seurin, and P. Wuille, "MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces." Cryptology ePrint Archive, Report 2020/1057, 2020. https://eprint.iacr.org/2020/1057.
10. J. Bootle, A. Cerulli, P. Chaidos, J. Groth, and C. Petit, "Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting," in *EUROCRYPT 2016, Part II* (M. Fischlin and J.-S. Coron, eds.), vol. 9666 of *LNCS*, pp. 327–357, Springer, Heidelberg, May 2016.
11. J. Jaeger and S. Tessaro, "Expected-time cryptography: Generic techniques and applications to concrete soundness," in *TCC 2020, Part III* (R. Pass and K. Pietrzak, eds.), vol. 12552 of *LNCS*, pp. 414–443, Springer, Heidelberg, Nov. 2020.
12. A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *CRYPTO'86* (A. M. Odlyzko, ed.), vol. 263 of *LNCS*, pp. 186–194, Springer, Heidelberg, Aug. 1987.
13. J. Groth, "On the size of pairing-based non-interactive arguments," in *EUROCRYPT 2016, Part II* (M. Fischlin and J.-S. Coron, eds.), vol. 9666 of *LNCS*, pp. 305–326, Springer, Heidelberg, May 2016.
14. E. Ben-Sasson, A. Chiesa, and N. Spooner, "Interactive oracle proofs," in *TCC 2016-B, Part II* (M. Hirt and A. D. Smith, eds.), vol. 9986 of *LNCS*, pp. 31–60, Springer, Heidelberg, Oct. / Nov. 2016.
15. J. Holmgren, "On round-by-round soundness and state restoration attacks." Cryptology ePrint Archive, Report 2019/1261, 2019. https://eprint.iacr.org/2019/1261.
16. Y. Lindell, "Parallel coin-tossing and constant-round secure two-party computation," in *CRYPTO 2001* (J. Kilian, ed.), vol. 2139 of *LNCS*, pp. 171–189, Springer, Heidelberg, Aug. 2001.
17. J. Groth and Y. Ishai, "Sub-linear zero-knowledge argument for correctness of a shuffle," in *EUROCRYPT 2008* (N. P. Smart, ed.), vol. 4965 of *LNCS*, pp. 379–396, Springer, Heidelberg, Apr. 2008.
18. M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, "Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings," in *ACM CCS 2019* (L. Cavallaro, J. Kinder, X. Wang, and J. Katz, eds.), pp. 2111–2128, ACM Press, Nov. 2019.
19. R. S. Wahby, I. Tzialla, a. shelat, J. Thaler, and M. Walfish, "Doubly-efficient zkSNARKs without trusted setup," in *2018 IEEE Symposium on Security and Privacy*, pp. 926–943, IEEE Computer Society Press, May 2018.

20. J. Lee, "Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments," *Cryptology ePrint Archive:2020/1274*, 2020.
21. B. Bünz, B. Fisch, and A. Szepieniec, "Transparent SNARKs from DARK compilers," in *EUROCRYPT 2020, Part I* (A. Canteaut and Y. Ishai, eds.), vol. 12105 of *LNCS*, pp. 677–706, Springer, Heidelberg, May 2020.
22. A. Chiesa, Y. Hu, M. Maller, P. Mishra, N. Vesely, and N. P. Ward, "Marlin: Preprocessing zkSNARKs with universal and updatable SRS," in *EUROCRYPT 2020, Part I* (A. Canteaut and Y. Ishai, eds.), vol. 12105 of *LNCS*, pp. 738–768, Springer, Heidelberg, May 2020.
23. R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. N. Rothblum, and R. D. Rothblum, "Fiat-Shamir from simpler assumptions." Cryptology ePrint Archive, Report 2018/1004, 2018. https://eprint.iacr.org/2018/1004.
24. R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. N. Rothblum, R. D. Rothblum, and D. Wichs, "Fiat-Shamir: from practice to theory," in *51st ACM STOC* (M. Charikar and E. Cohen, eds.), pp. 1082–1090, ACM Press, June 2019.
25. C. Lund, L. Fortnow, H. J. Karloff, and N. Nisan, "Algebraic methods for interactive proof systems," in *31st FOCS*, pp. 2–10, IEEE Computer Society Press, Oct. 1990.
26. I. Haitner, "A parallel repetition theorem for any interactive argument," in *50th FOCS*, pp. 241–250, IEEE Computer Society Press, Oct. 2009.
27. J. Håstad, R. Pass, D. Wikström, and K. Pietrzak, "An efficient parallel repetition theorem," in *TCC 2010* (D. Micciancio, ed.), vol. 5978 of *LNCS*, pp. 1–18, Springer, Heidelberg, Feb. 2010.
28. K.-M. Chung and F.-H. Liu, "Parallel repetition theorems for interactive arguments," in *TCC 2010* (D. Micciancio, ed.), vol. 5978 of *LNCS*, pp. 19–36, Springer, Heidelberg, Feb. 2010.
29. I. Berman, I. Haitner, and E. Tsfadia, "A tight parallel repetition theorem for partially simulatable interactive arguments via smooth KL-divergence," in *CRYPTO 2020, Part III* (D. Micciancio and T. Ristenpart, eds.), vol. 12172 of *LNCS*, pp. 544–573, Springer, Heidelberg, Aug. 2020.
30. E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward, "Aurora: Transparent succinct arguments for R1CS," in *EUROCRYPT 2019, Part I* (Y. Ishai and V. Rijmen, eds.), vol. 11476 of *LNCS*, pp. 103–128, Springer, Heidelberg, May 2019.
31. E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable zero knowledge with no trusted setup," in *CRYPTO 2019, Part III* (A. Boldyreva and D. Micciancio, eds.), vol. 11694 of *LNCS*, pp. 701–732, Springer, Heidelberg, Aug. 2019.
32. B. Bünz, M. Maller, P. Mishra, N. Tyagi, and P. Vesely, "Proofs for inner pairing products and applications," *Cryptology ePrint Archive:2019/1177*, 2020.
33. G. Fuchsbauer, A. Plouviez, and Y. Seurin, "Blind schnorr signatures and signed ElGamal encryption in the algebraic group model," in *EUROCRYPT 2020, Part II* (A. Canteaut and Y. Ishai, eds.), vol. 12106 of *LNCS*, pp. 63–95, Springer, Heidelberg, May 2020.
34. A. Ghoshal and S. Tessaro, "Tight state-restoration soundness in the algebraic group model." Cryptology ePrint Archive, Report 2020/1351, 2020. https://eprint.iacr.org/2020/1351.