Simulation Extractable Versions of Groth's zk-SNARK Revisited

Karim Baghery[1], Zaira Pindado[2] and Carla Ràfols[2]

[1] imec-COSIC, KU Leuven, Leuven, Belgium,
karim.baghery@kuleuven.be
[2] Universitat Pompeu Fabra, Barcelona, Spain,
zaira.pindado@upf.edu,carla.rafols@upf.edu

Abstract. Among various Non-Interactive Zero-Knowledge (NIZK) arguments, zk-SNARKs are the most efficient in terms of proof size and verification, which are two important criteria for large scale applications. Currently, Groth's construction from Eurocrypt'16, Groth16, is the most efficient and widely deployed one. However, it is proven to achieve only knowledge soundness, which does not prevent attacks from the adversaries who have seen simulated proofs. There has been considerable progress in modifying Groth16 to achieve simulation extractability to guarantee the non-malleability of proofs.
We revise the Simulation Extractable version of Groth16 proposed by Bowe and Gabizon in the Random Oracle Model, the most efficient one in terms of prover efficiency and common reference string size among the candidates. We present two variations of their construction which require 4 pairings in the verification, instead of 5. The first one has the same performance as Bowe and Gabizon's in all other parameters. The second one gets rid of the Random Oracle at the cost of a collision-resistant hash function, a single new element in the common reference string, and one exponentiation in the target group for the verifier. Both of our variants are among the most efficient simulation extractable versions of Groth16 in most dimensions.

Keywords: NIZK, zk-SNARK, Simulation Extractability, Generic Group Model

1 Introduction

Non-Interactive Zero-Knowledge (NIZK) proof systems are a fundamental family of cryptographic primitives that has appeared recently in a wide range of practical applications. A NIZK proof system allows a party to prove that for a public statement $\vec{x}$, she knows a witness $\vec{w}$ such that $(\vec{x}, \vec{w}) \in \mathbf{R}$, for some relation $\mathbf{R}$, without leaking any information about $\vec{w}$ and without interaction with the verifier. Due to their impressive advantages, NIZK proof systems are used ubiquitously to build larger cryptographic protocols and systems.

Zero-knowledge Succinct Arguments of Knowledge (zk-SNARKs) are among the most interesting NIZK proof systems in practice, as they allow to generate very short proofs for NP complete languages and, consequently, they are also very efficient to verify ([15, 17]). zk-SNARKs have had a tremendous impact in cryptographic practice and they have found numerous applications, including verifiable computation systems [22], privacy-preserving (PP) cryptocurrencies [7], PP smart contract systems [20], PP proof-of-stake

protocols [19], and efficient ledger verification protocols [10], are some of the best known applications that use zk-SNARKs to prove different statements very efficiently while guaranteeing the privacy of the prover. Because of their practical importance, particularly in large-scale applications like blockchains, even minimal savings (especially in proof size or verification cost) are considered to be relevant.

In 2016, Groth [17] introduced the most efficient zk-SNARK for Quadratic Arithmetic Programs or QAPs, which is still the state-of-the-art, Groth16. Its proof is 3 group elements and the cost of verification is dominated by 3 pairing computations. In the original paper, it is proven to achieve knowledge soundness in the generic group model (GGM). The proof of Groth16 is malleable, as it is shown in [18]. Generating non-malleable proofs is a necessary requirement in building various cryptographic schemes, including *universally composable* protocols [20, 19], cryptocurrencies (e.g. Zcash) [7], signature-of-knowledge schemes [18], etc. Therefore, in practice, it is important to have a stronger notion of knowledge soundness, known as (strong) simulation extractability (SE). This notion guarantees that a valid witness can be extracted from any adversary producing a proof accepted by the verifier, even after seeing an arbitrary number of simulated proofs.

There have been considerable efforts to refine Groth's zk-SNARK to achieve SE and guarantee the non-malleability of proofs. Firstly, in 2017 Groth and Maller [18] proposed a SE zk-SNARK, which is very efficient in terms of proof size but very inefficient in terms of Common Reference String (crs) size and prover time. They also showed how one can use SE zk-SNARKs to build Signature of Knowledge (SoK) schemes [13] with *succinct* signatures. In 2018 Bowe and Gabizon [11] proposed a less efficient construction in terms of proof size (5 group elements vs 3 in the original version) based on Groth16 which needs a Random Oracle (RO) (apart from GGM), but with almost no overhead in the crs size or additional cost for the prover. Last year, Lipmaa [21] proposed several constructions, including the most efficient QAP-based SE zk-SNARK in terms of proof size and with the same verification complexity as [18, 11], but less efficient in terms of crs size and prover time compared to [11]. In [2], Atapoor and Baghery used the traditional OR technique to achieve SE in Groth16. Their variant requires 1 paring less for verification in comparison with previous SE constructions, however it comes with an overhead in proof generation, crs, and even larger overhead in proof size. For a particular instantiation they add ≈ 52.000 constrains to the underlying QAP instance, which adds fixed overhead to the prover and crs, that can be considerable for mid-size circuits. They show that for a circuit with 10×10^6 Multiplication (Mul) gates, their prover is about 10% slower, but it can be slower for circuits with less than 10×10^6 gates.

Recently, Baghery, Kohlweiss, Siim, and Volkhov [6] explore another direction. Instead of modifying Groth16 to achieve strong SE, they first show that the original construction of Groth16 achieves weak SE with white-box extraction. Weak SE allows proof randomization, while it guarantees that a proof cannot be changed to prove a new statement. Then, considering the first result, they propose two efficient constructions of Groth16 that achieve weak SE with *black-box* extraction which is shown to be necessary for UC-security. Both *weak* and *strong* SE zk-SNARKs can be lifted to achieve black-box simulation extractability with a simple compiler [3, 6]. However, to realize the standard ideal functionality defined for NIZK arguments, one would need to use a strong SE NIZK with black-box extraction [16].

Table 1. A comparison of our proposed variations of Groth16 along with the other SE zk-SNARKs for arithmetic circuit satisfiability with n Mul gates (constraints) and m wires (variables), of which l are public input wires (variables). A typical set of values is $n = m = 10^6$ and $l = 10$. In the case of crs size and prover's computation we omit constants. In [18], n Mul gates and m wires translate to $2n$ squaring gates and $2m$ wires. In [2], SE is achieved with an OR approach which requires to add constraints and variables, resulting in $n' \approx n + 52.000$, $m' \approx m + 52.000$, and $l' = l + 4$. $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$: group elements, E_i: exponentiation in group $\mathbb{G}_i$, M_i: multiplication in group $\mathbb{G}_i$, P: pairings. GGM: Generic Group Model, ROM: Random Oracle Model, AGM: Algebraic Group Model, CRH: Collision Resistant Hash.

SNARK	Security	Model	crs	Prover	Proof	Verifier
Groth16 [17]	Knowledge Sound	GGM	$m + 2n - l$ $\mathbb{G}_1$ n $\mathbb{G}_2$	$m + 3n - l$ E_1 n E_2	2 $\mathbb{G}_1$ 1 $\mathbb{G}_2$	l E_1 3 P
GM [18]	Simulation Extractable	GGM	$2m + 4n$ $\mathbb{G}_1$ $2n$ $\mathbb{G}_2$	$2m + 4n - l$ E_1 $2n$ E_2	2 $\mathbb{G}_1$ 1 $\mathbb{G}_2$	l E_1 5 P
BG [11]	Simulation Extractable	GGM, ROM	$m + 2n - l$ $\mathbb{G}_1$ n $\mathbb{G}_2$	$m + 3n - l$ E_1 n E_2	3 $\mathbb{G}_1$ 2 $\mathbb{G}_2$	l E_1 5 P
AB [2]	Simulation Extractable	GGM	$m' + 2n' - l$ $\mathbb{G}_1$ n' $\mathbb{G}_2$	$m' + 3n' - l$ E_1 n' E_2	4 $\mathbb{G}_1$ 2 $\mathbb{G}_2$ + 2 λ	$l' + 2$ E_1 4 P
Lipmaa [21]	Simulation Extractable	AGM, Tag-based	$m + 3n - l$ $\mathbb{G}_1$ n $\mathbb{G}_2$	$m + 4n - l$ E_1 n E_2	3 $\mathbb{G}_1$ 1 $\mathbb{G}_2$	$l + 1$ E_1 5 P
Section 3	Simulation Extractable	GGM, ROM	$m + 2n - l$ $\mathbb{G}_1$ n $\mathbb{G}_2$	$m + 3n - l$ E_1 n E_2	3 $\mathbb{G}_1$ 2 $\mathbb{G}_2$	l E_1, 1 E_2 4 P
Section 4	Simulation Extractable	GGM, CRH	$m + 2n - l$ $\mathbb{G}_1$ n $\mathbb{G}_2$	$m + 3n - l$ E_1 n E_2	3 $\mathbb{G}_1$ 2 $\mathbb{G}_2$	l E_1, 1 E_2 1 E_T, 4 P

1.1 Our Contributions

In this work, we revise the simulation extractable variants of Groth16, presented in [11] and [2], to get the best of both constructions.

Our focus is mainly on Bowe and Gabizon's variation [11] which has the most efficient prover and the shortest crs among other SE zk-SNARKs [18, 11, 21, 2], while requires a RO. To achieve (strong) simulation extractability, their prover replaces all the original computations which depend on some parameter δ given in the crs by some δ' and the prover must give $[\delta']_2$ and a proof of knowledge (PoK) of the DLOG of $[\delta']_2$ w.r.t $[\delta]_2$.

Using the same approach [11], we construct two *strong* SE zk-SNARKs that are the most efficient simulation extractable variants of Groth16 in terms of crs size, prover complexity, and verification. Both zk-SNARKs use some sophisticated modification of Boneh-Boyen signatures [9] to prove knowledge of the DLOG of δ' which require 1 pairing less in the verification in comparison with the argument in Bowe and Gabizon's construction. The first construction uses non-programmable RO, while in the second construction, in the cost of a single new element in the crs and a collision-resistant hash function, we get rid of the RO and similar to Groth16, prove the security of construction in the GGM model.

Tab. 1 presents a comparison of our proposed variants of Groth16 with several other constructions for a particular instance of arithmetic circuit satisfiability. As it can be seen, in comparison with [11], both our constructions require 1 paring less in the verification, while retaining all the other properties of their construction. The second construction avoids using ROs, in the cost of a single new element in the crs which is negligible in

practice. In comparison with [2], both of our variants have a negligible overhead in the proof generation and crs size, and they both also come with smaller overhead in proof size. [3] Among two proposed variants, both constructions require 4 parings in the verification, however considering the number of exponentiations, we expect to have a slightly faster verification in the first construction, presented in Section 3.

Finally, we highlight that using the technique proposed in [18], both the proposed SE zk-SNARKs can be used to build *succinct* SoK schemes, which would be more efficient than previous constructions. In general, due to relying on non-falsifiable assumptions, succinct SoK schemes have better efficiency in comparison with the constructions that are built under standard assumptions [13, 8, 5]. We also note that to achieve strong (white-box) SE, our proposed zk-SNARKs require minimal changes in comparison with the original Groth16, particularly the proof generation and proof verification of Groth16 is a part of the proof generation and verification in our protocols. Therefore, one can use the same compiler or ad-hoc approach proposed in [3] and [6], respectively, to construct a more efficient strong *black-box* SE zk-SNARK for UC-protocols [16].

1.2 Organization

In Section 2, we introduce notation, the relevant security definitions, and recall the Boneh-Boyen signature scheme. In Section 3, we give our first SE zk-SNARK from non-programmable RO in the GGM, and in Section 4 our second SE zk-SNARK in GGM without RO.

1.3 Novelty

This note is the full version of a short paper with the same title appearing in CANS 20. The first construction in Section 3 only appears in this full version.

2 Preliminaries

2.1 Notation and bilinear groups

We let BGgen be a probabilistic polynomial time algorithm which on input 1^λ, where λ is the security parameter, returns the description of an asymmetric bilinear group $\mathsf{gk} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathcal{P}_1, \mathcal{P}_2)$, where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of prime order p, the elements $\mathcal{P}_1, \mathcal{P}_2$ are generators of $\mathbb{G}_1, \mathbb{G}_2$ respectively, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable, non-degenerate bilinear map, and there is no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.

Elements in $\mathbb{G}_\gamma$, are denoted implicitly as $[a]_\gamma = a\mathcal{P}_\gamma$, where $\gamma \in \{1, 2, T\}$ and $\mathcal{P}_T = e(\mathcal{P}_1, \mathcal{P}_2)$. With this notation, $e([a]_1, [b]_2) = [ab]_T$. We extend this notation naturally to vectors and matrices. We denote by $\mathsf{negl}(\lambda)$ an arbitrary negligible function in λ.

[3] In the worst case, our changes add only one element to the crs of Groth16 and since Groth16 is already proven to achieve subversion ZK (ZK without trusting a third party) [1, 14], our variants also can be proven to achieve Sub-ZK using the technique proposed in [4].

2.2 Definitions

For algorithms $\mathcal{A}$ and $\mathcal{E}_{\mathcal{A}}$, we write $(y \,\|\, y') \leftarrow (\mathcal{A} \,\|\, \mathcal{E}_{\mathcal{A}})(x; r)$ as a shorthand for "$y \leftarrow \mathcal{A}(x; r)$, $y' \leftarrow \mathcal{E}_{\mathcal{A}}(x; r)$". For an algorithm $\mathcal{A}$, let $\mathbf{Im}(\mathcal{A})$ be the image of $\mathcal{A}$, i.e. the set of valid outputs of $\mathcal{A}$, let $\mathsf{RND}(\mathcal{A})$ denote the random tape of $\mathcal{A}$. By $y \leftarrow \mathcal{A}(x; r)$ we denote the fact that $\mathcal{A}$, given an input x and a randomizer r, outputs y.

We use the definitions of NIZK arguments from [17]. Let $\mathcal{R}$ be a relation generator, such that $\mathcal{R}(1^\lambda)$ returns a polynomial-time decidable binary relation $\mathbf{R} = \{(\vec{x}, \vec{w})\}$. Here, $\vec{x}$ is the statement and $\vec{w}$ is the witness. Security parameter λ can be deduced from the description of $\mathbf{R}$. The relation generator also outputs auxiliary information $\mathsf{z}_{\mathbf{R}}$ that will be given to the honest parties and the adversary. As in [17], $\mathsf{z}_{\mathbf{R}}$ is the value returned by $\mathsf{BGgen}(1^\lambda)$, and is given as an input to the parties.

Let $\mathcal{L}_{\mathbf{R}} = \{\vec{x} : \exists \vec{w}, (\vec{x}, \vec{w}) \in \mathbf{R}\}$ be an NP-language. A *NIZK argument system* Ψ for $\mathcal{R}$ consists of tuple of PPT algorithms $(\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$, such that:

CRS Generator: K is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$ where $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \in \mathbf{Im}(\mathcal{R}(1^\lambda))$, outputs $\mathsf{crs} := (\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}})$ and stores trapdoors of crs as $\vec{\mathsf{ts}}$. We distinguish $\mathsf{crs}_{\mathsf{P}}$ (needed by the prover) from $\mathsf{crs}_{\mathsf{V}}$ (needed by the verifier).

Prover: P is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \vec{x}, \vec{w})$, where $(\vec{x}, \vec{w}) \in \mathbf{R}$, outputs an argument π. Otherwise, it outputs $\perp$.

Verifier: V is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x}, \pi)$, returns either 0 (reject) or 1 (accept).

Simulator: Sim is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}, \vec{\mathsf{ts}}, \vec{x})$, outputs a simulated argument π.

Besides *succinct* proofs, i.e. polynomial in λ, an SE zk-SNARK is required to satisfy *completeness*, *simulation extractability*, and *zero-knowledge*.

Definition 1 (Perfect Completeness). *A non-interactive argument Ψ is* perfectly complete for $\mathcal{R}$, *if for all λ, all $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \in \mathbf{Im}(\mathcal{R}(1^\lambda))$, and $(\vec{x}, \vec{w}) \in \mathbf{R}$,*

$$\Pr\left[\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}),\ \pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \vec{x}, \vec{w}) : \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x}, \pi) = 1 \quad \right] = 1.$$

Here, $\mathsf{z}_{\mathbf{R}}$ can be seen as a common auxiliary input to $\mathcal{A}$ that is generated by using a benign relation generator.

Definition 2 ((White-box) Simulation Extractability [18]). *Let $\mathsf{RND}(\mathcal{A})$ denote the random tape of $\mathcal{A}$. A non-interactive argument Ψ is* (strong) simulation-extractable for $\mathcal{R}$, *if for any NUPPT $\mathcal{A}$, there exists a NUPPT extractor $\mathsf{Ext}_{\mathcal{A}}$ s.t. for all λ,*

$$\Pr\begin{bmatrix}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \leftarrow \mathcal{R}(1^\lambda), (\mathsf{crs} \,\|\, \vec{\mathsf{ts}}) \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}), r \leftarrow_r \mathsf{RND}(\mathcal{A}), \\ ((\vec{x}, \pi) \,\|\, \vec{w}) \leftarrow (\mathcal{A}^{\mathsf{O}(\vec{\mathsf{ts}}, .)} \,\|\, \mathsf{Ext}_{\mathcal{A}})(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}; r) : \\ (\vec{x}, \pi) \notin Q \wedge (\vec{x}, \vec{w}) \notin \mathbf{R} \wedge \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x}, \pi) = 1\end{bmatrix} = \mathsf{negl}(\lambda).$$

Here, Q is the set of simulated statement-proof pairs. Note that *simulation extractability* implies *knowledge soundness*.

Definition 3 (Zero-Knowledge (ZK) [17]). *A non-interactive argument Ψ is* computationally ZK for $\mathcal{R}$, *if for all λ, all $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \in \mathbf{Im}(\mathcal{R}(1^\lambda))$, and for all NUPPT $\mathcal{A}$, $\varepsilon_0 \approx_c \varepsilon_1$, where*

$$\varepsilon_b = \Pr[(\mathsf{crs} \,\|\, \vec{\mathsf{ts}}) \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) : \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}) = 1].$$

Here, the oracle $\mathsf{O}_0(\vec{x}, \vec{w})$ returns $\perp$ (reject) if $(\vec{x}, \vec{w}) \notin \mathbf{R}$, and otherwise it returns $\mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \vec{x}, \vec{w})$. Similarly, $\mathsf{O}_1(\vec{x}, \vec{w})$ returns $\perp$ (reject) if $(\vec{x}, \vec{w}) \notin \mathbf{R}$, otherwise it returns $\mathsf{Sim}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}, \vec{\mathsf{ts}}, \vec{x})$. Ψ is perfect ZK for $\mathcal{R}$ *if one requires that $\varepsilon_0 = \varepsilon_1$.*

2.3 Boneh-Boyen signatures

We briefly recall Boneh-Boyen signatures [9]. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear group. Messages are elements of $\mathbb{Z}_p$, and signatures are elements of $\mathbb{G}_1$. The secret key is $\mathsf{sk} \in \mathbb{Z}_p$, and the public key (verification key) is $[\mathsf{sk}]_2 \in \mathbb{G}_2$. To sign a message $\mathsf{m} \in \mathbb{Z}_p$, the signer computes

$$[\sigma]_1 = \left[\frac{1}{\mathsf{sk} + \mathsf{m}}\right]_1.$$

The verifier accepts the signature if the equation $e([\sigma]_1, [\mathsf{sk}]_2 + [\mathsf{m}]_2) = [1]_T$ holds.

Boneh-Boyen signatures are existentially unforgeable under the q-SDH assumption. We use them in our constructions as proofs of knowledge of the secret key in the generic group model.

3 A Simulation Extractable zk-SNARK in the ROM

As we discussed, the main idea in Bowe and Gabizon's [11] work to achieve simulation extractability is to replace all the computations which depend on some parameter δ given in the crs by some randomization of it, say δ', and the prover must give $[\delta']_2$ and a Proof of Knowledge (PoK) in the ROM of the Discrete Logarithm (DLOG) of $[\delta']_2$ w.r.t $[\delta]_2$. This makes it harder for the adversary to re-use elements from the simulated proofs that are created with the original parameter δ.

Our idea is to replace the PoK with a Boneh-Boyen signature. A nice feature of this construction inherited from [11] is that SE is achieved essentially without modifications in the crs or the prover complexity, or changes in the security model (which is still Generic Group Model and Random Oracle Model).

3.1 Scheme definition

In Fig. 1, we describe the proposed variation of $\mathsf{Groth16}$ that can achieve SE. We highlight the changes in the new construction with gray background.

Our modification follows closely the one of Bowe and Gabizon [11], except that in their scheme $[d]_1 = [y]_1\zeta$ where $[y]_1 = H(A \parallel B \parallel C \parallel \delta')$ and their verification checks that $[\delta']_1[y]_2 = [d]_1[\delta]_2$, which requires 2 pairings. The security proof shows that this is a simulation extractable PoK of the DLOG of $[\delta']_2$ with respect to $[\delta]_2$. We follow the same idea but our approach embeds a Boneh-Boyen signature in the proof as argument of knowledge for this DLOG, which requires 1 pairing, instead of 2.

Setup, $\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$**:** Similar to the original scheme it picks $x, \alpha, \beta, \delta \leftarrow \mathbb{Z}_p^*$, $H \leftarrow \mathcal{H}$, and returns crs defined as the following (by considering the observation in [12] that γ in the original scheme can be set 1),

$$(\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}}) := \mathsf{crs} \leftarrow \begin{pmatrix} [\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \\ \left\{\frac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta}\right\}_{j=l+1}^{m}, \{x^i t(x)/\delta\}_{i=0}^{n-2}]_1, \\ [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2,\ [\alpha\beta, t(x)]_T, H \end{pmatrix}.$$

Prover, $\pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \vec{x} = (a_1, \ldots, a_l), \vec{w} = (a_{l+1}, \ldots, a_m))$**:** assuming $a_0 = 1$, it acts as follows,

1. Selects a random element $\zeta \leftarrow \mathbb{Z}_p^*$, and sets $[\delta']_2 := \zeta[\delta]_2$
2. Let $A^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j u_j(X)$, $B^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j v_j(X)$, $C^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j w_j(X)$,
3. Set $h(X) = \sum_{i=0}^{n-2} h_i X^i \leftarrow (A^\dagger(X)B^\dagger(X) - C^\dagger(X))/t(X)$,
4. Set $[h(x)t(x)/\delta]_1 \leftarrow \sum_{i=0}^{n-2} h_i [x^i t(x)/\delta']_1$,
5. Set $r_a \leftarrow_r \mathbb{Z}_p$; Set $[A]_1 \leftarrow \sum_{j=0}^{m} a_j [u_j(x)]_1 + [\alpha]_1 + r_a [\delta']_1$,
6. Set $r_b \leftarrow_r \mathbb{Z}_p$; Set $[B]_2 \leftarrow \sum_{j=0}^{m} a_j [v_j(x)]_2 + [\beta]_2 + r_b [\delta']_2$,
7. Set $[C]_1 \leftarrow r_b[A]_1 + r_a \left(\sum_{j=0}^{m} a_j [v_j(x)]_1 + [\beta]_1\right) + \sum_{j=l+1}^{m} a_j [(u_j(x)\beta + v_j(x)\alpha + w_j(x))/\delta']_1 + [h(x)t(x)/\delta']_1$,
8. Sets $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$, where $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a secure hash function,
9. Computes $[D]_1 = \frac{1}{\zeta + m}[t(x)/\delta]_1 = [\frac{t(x)}{\delta' + m\delta}]_1$
10. Return $\pi := ([A, C, D]_1, [B, \delta']_2)$.

Verifier, $\{1, 0\} \leftarrow \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x} = (a_1, \ldots, a_l), \pi = ([A, C, D]_1, [B, \delta']_2))$**:** assuming $a_0 = 1$, and setting $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$ checks if

1. $[A]_1[B]_2 = [C]_1[\delta']_2 + \left(\sum_{j=0}^{l} a_j \left[u_j(x)\beta + v_j(x)\alpha + w_j(x)\right]_1\right)[1]_2 + [\alpha\beta]_T$
2. $[D]_1[\delta' + \delta m]_2 = [t(x)]_T$ (Note that:$[t(x)/\delta]_1[\delta]_2 = [t(x)]_T$)

and return 1 if both checks pass, otherwise return 0.

Simulator, $\pi \leftarrow \mathsf{Sim}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x} = (a_1, \ldots, a_l), \vec{\mathsf{ts}})$**:** Given the simulation trapdoors $\vec{\mathsf{ts}} := (\beta, \delta)$ acts as follows,

1. Choose random $\zeta \leftarrow_r \mathbb{Z}_p^*$ and set $\delta' := \zeta\delta$
2. Choose $A, B \leftarrow_r \mathbb{Z}_p$
3. Let $[C]_1 = \left[(A \cdot B - \sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x)) - \alpha\beta)/\delta'\right]_1$
4. Let $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$
5. Set $[D]_1 = \dfrac{t(x)}{\delta' + m\delta}[1]_1$
6. Return $\pi := ([A]_1, [B]_2, [C]_1, [D]_1, [\delta']_2)$

Fig. 1. The proposed simulation-extractable variation of $\mathsf{Groth16}$ for $\mathbf{R}$ along with a Boneh-Boyen signature. $\mathcal{H}$ is a family of collision resistant hash functions that map to $\mathbb{Z}_p^*$. The element $[t(x)]_T$ is redundant and can be computed from the rest of the elements in the crs. Alternatively, one can describe $\mathsf{Groth16}$ as corresponding to $\zeta = 1$ and where the proof consists only of $[A, C]_1, [B]_2$.

3.2 Security

A part from saving one pairing on verification with respect to [11], our scheme also has the nice property that the RO maps to elements in $\mathbb{Z}_p$ and it does not need the property that H can sample elements of $\mathbb{G}$ obliviously (i.e. soundness does not use that the DLOG of image elements is hard).

In a nutshell, we show that we can embed a Boneh-Boyen signature in the proof and this results in a SE argument of knowledge in the GGM and the ROM. Namely, the element $[1/(\delta' + \delta m)]_1$, which is a Boneh-Boyen signature of δm for public key $[\delta']_2$ can be constructed from $[1/\delta]_1$ for all $m \in \mathbb{Z}_p$, if and only if, the DLOG of δ' w.r.t δ is known. The adversary might be able to cheat for a specific m (i.e. if it sets $\delta' = k\delta - m^*\delta$ it can cheat for m^*) but the RO ensures that δ' cannot be set as a function of m. Given knowledge of the DLOG of δ', following the same blueprint as the proof of Bowe and Gabizon, we prove that the simulated queries are useless to the adversary. Then, we can easily conclude that the scheme is SE if $\mathsf{Groth16}$ is knowledge sound.

Theorem 1 (Completeness, ZK, SE). *The variation of $\mathsf{Groth16}$ described in Fig. 1, guarantees 1) perfect completeness, 2) perfect zero-knowledge and 3) simulation-extractability in the asymmetric Generic Group Model and the Random Oracle Model.*

Proof. Perfect completeness and perfect zero-knowledge are obvious and the proof is omitted. Knowledge extractability is proven by reduction (in the GGM) to the knowledge soundness of $\mathsf{Groth16}$. The reduction works in two steps (similarly to [11], although the proof of each of these steps is different):

Step 1 Extraction of the DLOG of δ'.
Step 2 Reduction to the Knowledge Soundnesss of $\mathsf{Groth16}$.

Proof of Step 1) Suppose $\mathcal{A}$ has made a sequence of queries $\vec{x}_1, \cdots, \vec{x}_v$ to $\mathsf{Sim}(\vec{\mathsf{ts}}, \cdot)$, and received answers $\{\pi_j = (A_j, B_j, C_j, D_j, \delta_j)\}_{j=1}^{v}$. Let Q' be the union of elements in the crs together with those from the replies of $\mathsf{Sim}(\vec{\mathsf{ts}}, \cdot)$; namely,

$$Q' := \begin{pmatrix} \big[\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \\ \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \\ \left\{\dfrac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta}\right\}_{j=l+1}^{m}, \\ \{x^i t(x)/\delta\}_{i=0}^{n-2}\big]_1, [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2 \end{pmatrix} \cup \begin{pmatrix} \left\{\left[A_j, C_j := \frac{A_j B_j - \mathsf{ic}_j - \alpha\beta}{\delta_j}, \right.\right. \\ \left.\left. D_j := \dfrac{t(x)}{\delta_j + m_j\delta}\right]_1 \right. \\ \left. [B_j, \delta_j]_2, m_j\right\}_{j=1}^{v} \end{pmatrix}$$

where $\mathsf{ic}_j = \sum_{i=0}^{l} a_i^j(u_i(x)\beta + v_i(x)\alpha + w_i(x))$, $\vec{x}_j = (a_1^j, \ldots, a_l^j)$, and $m_j \in \mathbb{Z}_p$ the message that simulator receives from the RO for each A_j, B_j, C_j, δ_j.

Now, assume $\mathcal{A}$ has produced elements (A, B, C, D, δ') such that

$$A \cdot B \equiv C \cdot \delta' + \left(\textstyle\sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x))\right) + \alpha\beta$$

and, for $m := H(A \parallel B \parallel C \parallel \delta')$, $D(\delta' + \delta m) = t(x)$. Let Q'_1 be the set with the elements of Q' in $\mathbb{G}_1$ and Q'_2 the elements in $\mathbb{G}_2$. Since the adversary is generic it has constructed these elements as a linear combination of the elements in Q' which are in the relevant

group (i.e. element of Q_1' in $\mathbb{G}_1$ for A, C, D and of Q_2' for B, δ') and we can extract the coefficients of this linear combination.

First, we prove that the adversary has knowledge of the discrete logarithm of δ' w.r.t. δ. From the second verification equation, $D = \frac{t(x)}{\delta' + \delta m}$. On the other hand, from adversary $\mathcal{A}$ we can recover a vector $\vec{k}_D$ with the coefficients that it has used to construct D, that is, $D = \sum_{q \in Q_1'} k_{D,q}\ q$, and a vector $\vec{k}_{\delta'}$ with the coefficients that it has used to construct $\delta' = \sum_{q \in Q_2'} k_{\delta',q}\ q$.

We argue that $\delta' + \delta m$ cannot be a polynomial in x, i.e. $\delta' + \delta m$ is a linear combination of terms in Q_1 without x. Indeed, if $\delta' + \delta m$ is a polynomial in x, then this polynomial must divide $t(x)$ because the adversary does not see any rational functions with x. Then, there exists a polynomial ν such that $\delta' + \delta m = (x - r)\nu$, for some r root of $t(x)$. However, since $\delta' + \delta m$ cannot have any terms $x\delta$ ($x\delta \notin Q_2'$), the only possibility is that ν does not have any term with δ, and neither $\delta' + \delta m$. This means that $\delta' = \delta'' - \delta m$, for some δ'' independent of δ. But since H is a RO, the probability that given δ' and δ'', m satisfies this relation, is $1/p$.

Therefore, x only appears in the numerator of the expression $D = \frac{t(x)}{\delta' + \delta m}$, and thus, we have

$$\frac{t(x)}{\delta' + \delta m} = k_{D,0} \frac{t(x)}{\delta} + \sum_{j=1}^{v} k_{D,j} \frac{t(x)}{\delta_j + m_j} \tag{1}$$

where, to simplify the notation, we define $k_{D,0} = k_{D, \frac{t(x)}{\delta}}$. $k_{D,j} = k_{D, \frac{t(x)}{\delta_j + m_j \delta}}$. Defining $\delta_0 = \delta$, $m_0 = m$, then

$$\frac{t(x)}{\delta' + \delta m} = \sum_{j=0}^{v} k_{D,j} \frac{t(x)}{\delta_j + m_j \delta} \iff \frac{1}{\delta' + \delta m} = \sum_{j=0}^{v} k_{D,j} \frac{\prod_{i=0, i \neq j}^{v} (\delta_i + m_i \delta)}{\prod_{i=0}^{v} (\delta_i + m_i \delta)} \tag{2}$$

$$\iff \prod_{i=0}^{v} (\delta_i + m_i \delta) = (\delta' + \delta m) \Big(\sum_{j=0}^{v} k_{D,j} \prod_{i=0, i \neq j}^{v} (\delta_i + m_i \delta) \Big). \tag{3}$$

It follows that the term $\delta' + m\delta$ must divide the left side of the equation (2). Therefore, there exists some index j^* and $k \in \mathbb{Z}_p$ such that $\delta' + m\delta = k(\delta_{j^*} + m_{j^*}\delta)$. Now, dividing Eq. (2) by $(\delta_{j^*} + m_{j^*}\delta)$, we come to the following expression

$$\textstyle\prod_{i=0, i \neq j^*}^{v} (\delta_i + m_i \delta) = k \cdot \Big(k_{D,j^*} \prod_{i=0, i \neq j^*}^{v} (\delta_i + m_i \delta) + \sum_{j=0, j \neq j^*}^{v} k_{D,j} \prod_{i=0, i \neq j}^{v} (\delta_i + m_i \delta) \Big),$$

which is equivalent to

$$0 = (1 - k \cdot k_{D,j^*}) \textstyle\prod_{i=0, i \neq j^*}^{v} (\delta_i + m_i \delta) - \sum_{j=0, j \neq j^*}^{v} k \cdot k_{D,j} \prod_{i=0, i \neq j}^{v} (\delta_i + m_i \delta).$$

Since all summands are linearly independent polynomials, $k = k_{D,j^*}^{-1}$, and $k_{D,j} = 0$ if $j \neq j^*$. We distinguish two cases: (1) $\delta' + m\delta = k\delta$ ($j^* = 0$) or (2) $\delta' + m\delta = k(\delta_{j^*} + m_{j^*}\delta)$ ($j^* \neq 0$).

In case (1), we are done, as we can extract the DLOG of δ' as $k - m$.

In case (2), there exists some $k' \in \mathbb{Z}_p$ such that $\delta' = k\delta_{j^*} + k'\delta$ and $m = km_{j^*} - k'$. Since H is a RO, m is a uniform random element given δ', (and thus, given k, k',) and therefore the probability of this event is $1/p$.

Thus, the adversary cannot compute the elements of such a proof belonging to the $\text{Span}(Q')$ unless it knows ζ.

Proof of Step 2) We show that the elements A, B, C do not use the elements of the simulated proofs, say $V := \{[A_j]_1, [B_j]_2, [C_j]_1, [D_j]_1, [\delta_j]_2\}_{j=1}^{v}$, and then, with the knowledge of ζ such that $\delta' = \zeta\delta$, we can reduce our proof to the knowledge soundness proof of Groth16 [17], since $[A]_1, [B]_2, [C\zeta]_1$ is a valid proof of it.

For this, we need to argue that A, B, C cannot have been constructed from any of the elements of the queries. To prove that A, B, C are not constructed from the elements $[A_j]_1, [B_j]_2, [C_j]_1, [\delta_j]_2$, we follow the exact same reasoning as Bowe and Gabizon [11] in the GGM and we omit the details. Next, we prove that to construct A, C the prover cannot have used any of the D_j terms, which are the new elements in our proof.

Assume A has been generated from some D_j, so the term $\frac{t(x)}{\delta_j + m_j\delta}$ appears in the expression of A generated from Q'_1 with the corresponding coefficient different than 0. Observe that the verification equation contains the term $\alpha\beta$ that cannot be manipulated because it is fixed in the crs, and it should be produced by the term AB because $\beta \in Q'_2$ and β is independent of $\delta' = \zeta\delta$. In that case, the product AB would contain a term $\frac{t(x)}{\delta_j + m_j\delta}\beta$, but this cannot be cancelled out by any of the other terms in the equation. Indeed, this term cannot appear in $\alpha\beta$, or in the sum of public values of a_i. Thus, the only possibility is that it appears in $C\delta'$. However, since β is independent of δ', it should appear in C, but $\frac{t(x)}{\delta_j + m_j\delta}\beta$ cannot be computed from elements in Q'_1.

Now, assume D_j appears in C, then the term $C\delta'$ of the verification includes $\frac{t(x)}{\delta_j + m_j\delta}\delta'$. Neither the term $\alpha\beta$ nor the sum of public values can include it, so the only possibility is that it appears in AB. Since $\delta' \in Q'_2$, then A would contain D_j, which we ruled out previously.

Note that we make the proof in the asymmetric GGM for simplicity, but the analogous proof in the symmetric model gives very similar impossible terms. The difference is that in the second part of the proof we have to analyse more possible cases (considering α, β in any of the groups).

4 A Simulation Extractable zk-SNARK without RO

In this section, we present another variation of the Groth16 which is very similar to the construction in Section 3. It also offers simulation extractability in the Generic Group Model, but without involving the Random Oracle. This is done in exchange for adding one element in the crs.

4.1 Scheme definition

In Fig. 2, we propose our second variation of Groth16 for QAP. It is inspired by the simulation extractable version of Bowe and Gabizon [11] and is very similar to the first construction of this work (see Section 3 for intuition).

In this approach, we change the Proof of Knowledge (PoK) of the DLOG of $[\delta']_2$ w.r.t. $[\delta]_2$ to another PoK in the GGM without using random oracles with a variation of Boneh-Boyen signatures, where we just use the collision resistance property of the hash function. We briefly give an intuition in the following.

Setup, $\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$**:** Similar to the original scheme pick $x, \alpha, \beta, \delta \leftarrow \mathbb{Z}_p^*$, $H \leftarrow \mathcal{H}$, and returns crs defined as the following,

$$(\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}}) := \mathsf{crs} \leftarrow \begin{pmatrix} \left[\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \frac{\gamma t(x)}{\delta}\right. \\ \left\{\frac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta}\right\}_{j=l+1}^{m}, \left\{\frac{x^i t(x)}{\delta}\right\}_{i=0}^{n-2}\Big]_1, \\ [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2,\ [\alpha\beta, t(x), \gamma t(x)]_T, H \end{pmatrix}.$$

Prover, $\pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \vec{x} = (a_1, \ldots, a_l), \vec{w} = (a_{l+1}, \ldots, a_m))$**:** assuming $a_0 = 1$, it acts as follows,

1. Selects a random element $\zeta \leftarrow \mathbb{Z}_p^*$, and sets $[\delta']_2 := \zeta[\delta]_2$
2. Let $A^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j u_j(X)$, $B^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j v_j(X)$,
3. Let $C^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j w_j(X)$,
4. Set $h(X) = \sum_{i=0}^{n-2} h_i X^i \leftarrow (A^\dagger(X)B^\dagger(X) - C^\dagger(X))/t(X)$,
5. Set $[h(x)t(x)/\delta']_1 \leftarrow (1/\zeta)(\sum_{i=0}^{n-2} h_i [x^i t(x)/\delta]_1)$,
6. Set $r_a \leftarrow_r \mathbb{Z}_p$; Set $[A]_1 \leftarrow \sum_{j=0}^{m} a_j [u_j(x)]_1 + [\alpha]_1 + r_a [\delta']_1$,
7. Set $r_b \leftarrow_r \mathbb{Z}_p$; Set $[B]_2 \leftarrow \sum_{j=0}^{m} a_j [v_j(x)]_2 + [\beta]_2 + r_b [\delta']_2$,
8. Set $[C]_1 \leftarrow r_b[A]_1 + r_a \left(\sum_{j=0}^{m} a_j [v_j(x)]_1 + [\beta]_1\right) +$
   $(1/\zeta) \sum_{j=l+1}^{m} a_j([(u_j(x)\beta + v_j(x)\alpha + w_j(x))/\delta]_1) + [h(x)t(x)/\delta']_1$,
9. Set $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$, where $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a secure hash function,
10. Compute $[D]_1 = \frac{m}{\zeta+m}[\frac{t(x)}{\delta}]_1 + \frac{1}{\zeta+m}[\frac{\gamma t(x)}{\delta}]_1 = [\frac{(m+\gamma)t(x)}{\delta'+m\delta}]_1$
11. Return $\pi := ([A, C, D]_1, [B, \delta']_2)$.

Verifier, $\{1, 0\} \leftarrow \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x} = (a_1, \ldots, a_l), \pi = ([A, C, D]_1, [B, \delta']_2))$**:** assuming $a_0 = 1$, and setting $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$ checks if

1. $[A]_1[B]_2 = [C]_1[\delta']_2 + \left(\sum_{j=0}^{l} a_j \left[u_j(x)\beta + v_j(x)\alpha + w_j(x)\right]_1\right)[1]_2 + [\alpha\beta]_T$
2. $[D]_1[\delta' + \delta m]_2 = m[t(x)]_T + [\gamma t(x)]_T$

and returns 1 if both checks pass, otherwise return 0.

Simulator, $\pi \leftarrow \mathsf{Sim}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \vec{x} = (a_1, \ldots, a_l),$ **):** Given the simulation trapdoors $\vec{\mathsf{ts}} := (\beta, \delta)$ acts as follows,

1. Choose random $\zeta \leftarrow_r \mathbb{Z}_p^*$ and set $\delta' := \zeta\delta$
2. Choose $A, B \leftarrow_r \mathbb{Z}_p$
3. Let $[C]_1 = [(A \cdot B - \sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x)) - \alpha\beta)/\delta']_1$
4. Let $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$
5. $[D]_1 = \frac{m}{\zeta+m}[\frac{t(x)}{\delta}]_1 + \frac{1}{\zeta+m}[\frac{\gamma t(x)}{\delta}]_1 = [\frac{(m+\gamma)t(x)}{\delta'+m\delta}]_1$
6. Return $\pi := ([A, C, D]_1, [B, \delta']_2)$.

Fig. 2. The proposed simulation-extractable variation of Groth16 for $\mathbf{R}$ along with a modification of the Boneh Boyen signature. $\mathcal{H}$ is a family of collision resistant hash functions that map to $\mathbb{Z}_p^*$. The elements $[\alpha\beta, t(x), \gamma t(x)]_T$ are redundant and can in fact be computed from the rest of the elements in the crs. Alternatively, one can describe Groth16 as corresponding to $\zeta = 1, \gamma = 0$ and where the proof consists only of $[A, C]_1, [B]_2$. Differences with Groth16 are highlighted.

Avoiding Random Oracle. Our proof uses the collision resistance property of the hash function and the generic group model. Very roughly, the new variable γ gives some

additional guarantees because to compute $t(x)\frac{(\gamma+m)}{(\delta'+\delta m)}$ from D_j such that $m_j \neq m$, it is necessary to know both $\frac{1}{(\delta'+\delta m)}$ and $\frac{\gamma}{(\delta'+\delta m)}$, but this is only possible when $\delta'+\delta m = k\delta$. Then, either we have the knowledge of the DLOG of δ' respect to δ $(k-m)$, which is straightforward, or either we have re-used δ'_j and m_j from some jth query. The last case is discarded when we reach that same message had to be re-used, $m = m_j$, which breaks collision resistance of the hash.

4.2 Security

We prove security of our construction (in Fig. 2) in the following theorem.

Theorem 2 (Completeness, ZK, SE). *The variation of Groth16 described in Fig. 2, guarantees (1) perfect completeness, 2) perfect zero-knowledge and 3) simulation-extractability in the asymmetric Generic Group Model.*

Proof. Perfect completeness and perfect zero-knowledge are obvious and the proof is omitted. Knowledge extractability is proven in the same way as the proof of Section 3 by reduction (in the GGM) to the knowledge soundness of Groth16, the reduction works in these two steps:

Step 1 Extraction of the DLOG of δ'.
Step 2 Reduction to the Knowledge Soundnesss of Groth16.

Proof of Step 1) Suppose $\mathcal{A}$ has made a sequence of queries $x_1, \cdots, x_v$ to $\mathsf{Sim}(\vec{\mathsf{ts}}, \cdot)$, and received answers $\{\pi_j = ([A_j]_1, [B_j]_2, [C_j]_1, [D_j]_1, [\delta_j]_2)\}_{j=1}^v$. Let Q' be the union of elements in the crs together with those from the replies of $\mathsf{Sim}(\vec{\mathsf{ts}}, \cdot)$; namely,

$$Q' := \begin{pmatrix} [\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \gamma t(x)/\delta \\ \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \\ \{\frac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta}\}_{j=l+1}^{m}, \\ \{x^i t(x)/\delta\}_{i=0}^{n-2}]_1, [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2 \end{pmatrix} \cup \begin{pmatrix} \Big\{\Big[A_j, C_j := \frac{A_j B_j - \mathsf{ic}_j - \alpha\beta}{\delta_j}, \\ D_j := \frac{t(x)(\gamma+m_j)}{\delta_j + m_j\delta}\Big]_1 \\ [B_j, \delta_j]_2, m_j\}_{j=1}^{v} \end{pmatrix}$$

where $\mathsf{ic}_j = \sum_{i=0}^{l} a_i^j(u_i(x)\beta + v_i(x)\alpha + w_i(x))$, $\vec{x}_j = (a_1^j, \ldots, a_l^j)$, and $m_j \in \mathbb{Z}_p$ the message that simulator receives from the hash function for each A_j, B_j, C_j, δ_j.

We assume the adversary $\mathcal{A}$ has produced elements (A, B, C, D, δ') such that

$$A \cdot B \equiv C \cdot \delta' + \left(\sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x))\right) + \alpha\beta$$

and, for $m := H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$, $D(\delta'+\delta m) = t(x)(m+\gamma)$. Let Q'_1 the elements in Q' in $\mathbb{G}_1$ and Q'_2 the elements in $\mathbb{G}_2$. Since the adversary is generic it has constructed these elements as a linear combination of the elements in Q' which are in the relevant group (i.e. element of Q'_1 in $\mathbb{G}_1$ for A, C, D and of Q'_2 for B, δ') and we can extract the coefficients of this linear combination.

First, we prove that the adversary has knowledge of the discrete logarithm of δ' w.r.t. δ. From the second verification equation we know that $D = t(x)\frac{\gamma+m}{\delta'+m\delta}$. On the other

hand, from adversary $\mathcal{A}$ we can recover a vector $\vec{k}_D$ with the coefficients that it has used to construct D, that is, $D = \sum_{q \in Q'_1} k_{D,q}\ q$. Equating these two expressions,

$$t(x)(m+\gamma) = (\textstyle\sum_{q \in Q'_1} k_{D,q}\ q)(\delta' + m\delta), \tag{4}$$

where $\delta' = \sum_{q \in Q'_2} k_{\delta',q}\ q$ for another vector of coefficients $\vec{k}_{\delta'}$. The terms which include γ in both sides of the equation must be the same.

On the other hand, by assumption, in the asymmetric GGM, the term δ' is constructed as a linear combination of elements in Q'_2 and therefore $\delta' + \delta m$ is independent of γ. Then, keeping only the terms with γ in equation (4), we obtain the following relation:

$$t(x)\gamma = k_{D,0}\frac{\gamma t(x)}{\delta}(\delta' + m\delta) + \textstyle\sum_{j=1}^{v} k_{D,j}\frac{\gamma t(x)}{\delta_j + m_j\delta}(\delta' + m\delta), \tag{5}$$

where we have set $k_{D,0} = k_{D,\frac{\gamma t(x)}{\delta}}$ and $k_{D,j} = k_{D,\frac{\gamma t(x)}{\delta_j + m_j\delta}}$ to simplify the notation.

Dividing both sides of the equation by $t(x)\gamma$ and defining $\delta_0 = \delta'$, $m_0 = 0$, we obtain the following equivalent equation:

$$1 = \left(\textstyle\sum_{j=0}^{v} k_{D,j}\frac{1}{\delta_j + m_j\delta}\right)(\delta' + m\delta) = \textstyle\sum_{j=0}^{v} k_{D,j}\frac{\prod_{i=0, i\neq j}^{v}(\delta_i + m_i\delta)}{\prod_{i=0}^{v}(\delta_i + m_i\delta)}(\delta' + m\delta)$$

$$\Leftrightarrow \textstyle\prod_{i=0}^{v}(\delta_i + m_i\delta) = (\delta' + m\delta)\left(\sum_{j=0}^{v} k_{D,j}\prod_{i=0, i\neq j}(\delta_i + m_i\delta)\right). \tag{6}$$

From the last equation it follows that the term $\delta' + m\delta$ must divide the left side of the equation (6). Therefore, there exists some index j^* and $k \in \mathbb{Z}_p$ such that $\delta' + m\delta = k(\delta_{j^*} + m_{j^*}\delta)$. Now, dividing Eq. (6) by $(\delta_{j^*} + m_{j^*}\delta)$, we come to the following expression

$$0 = (1 - k \cdot k_{D,j^*})\textstyle\prod_{i=0, i\neq j^*}^{v}(\delta_i + m_i\delta) - \sum_{j=0, j\neq j^*}^{v} k_{D,j}\prod_{i=0, i\neq j}^{v}(\delta_i + m_i\delta).$$

Since all summands are linearly independent polynomials, $k = k_{D,j^*}^{-1}$, and $k_{D,j} = 0$ if $j \neq j^*$. We distinguish two cases: (1) $\delta' + \delta m = k\delta$ ($j^* = 0$) or (2) $\delta' + \delta m = k(\delta_{j^*} + m_{j^*}\delta)$ ($j^* \neq 0$).

In case (1), we are done, as we can extract the DLOG of δ' as $k - m$.

In case (2), from equation (4) and putting everything together, we have that:

$$t(x)(m+\gamma) = k_{D,j^*}\frac{(\gamma + m_{j^*})}{(\delta_{j^*} + m_{j^*}\delta)}(\delta' + m\delta) = k_{D,j^*}k^{-1}(\gamma + m_{j^*})t(x) = (\gamma + m_{j^*})t(x).$$

This implies that $m_{j^*} = m$ is a collision of H.

Proof of Step 2) We show that the elements A, B, C do not use the elements of the simulated proofs, say $V := \{[A_j]_1, [B_j]_2, [C_j]_1, [D_j]_1, [\delta_j]_2\}_{j=1}^{v}$, and then, with the knowledge of ζ such that $\delta' = \zeta\delta$, we can reduce our proof to the knowledge soundness proof of Groth16 [17], since $[A]_1, [B]_2, [C\zeta]_1$ is a valid proof of Groth16.

For this, we need to argue that A, B, C cannot have been constructed from any of the elements of the queries. To prove that A, B, C are not constructed from the elements $[A_j]_1, [B_j]_2, [C_j]_1, [\delta_j]_2$, we follow the exact same reasoning as Bowe and Gabizon [11] in the GGM and we omit the details. Next, we prove that to construct A, C the prover cannot have used any of the D_j terms, which are the new elements in our proof.

Analogously to proof in Section 3, assume A has been generated from some D_j, so the term $\frac{t(x)(m_j+\gamma)}{\delta_j + m_j\delta}$ appears in the expression of A generated from Q'_1 with the corresponding

coefficient different than 0. Observe that the verification equation contains the term $\alpha\beta$ that cannot be manipulated because it is fixed in the crs, and it should be produced by the term AB because $\beta \in Q_2'$ and β is independent of $\delta' = \zeta\delta$. In that case, the product AB would contain a term $\frac{t(x)(m_j+\gamma)}{\delta_j+m_j\delta}\beta$, but this cannot be cancelled out by any of the other terms in the equation. Indeed, this term cannot appear in $\alpha\beta$, or in the sum of public values of a_i. Thus, the only possibility is that it appears in $C\delta'$. However, since β is independent of δ', it should appear in C, but $\frac{t(x)(m_j+\gamma)}{\delta_j+m_j\delta}\beta$ cannot be computed from elements in Q_1'.

Now, assume D_j appears in C, then the term $C\delta'$ of the verification includes $\frac{t(x)(m_j+\gamma)}{\delta_j+m_j\delta}\delta'$. Neither the term $\alpha\beta$ nor the sum of public values can include it, so the only possibility is that it appears in AB. Since $\delta' \in Q_2'$, then A would contain D_j, which we ruled out previously.

□

Acknowledgement. We thank Alonso González for his helpful discussions. Carla Ràfols and Zaira Pindado were partially supported by Project RTI2018-102112-B-I00 (AEI/FEDER, UE). Karim Baghery was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0085, and by Cyber Security Research Flanders with reference number VR20192203.

References

1. B. Abdolmaleki, K. Baghery, H. Lipmaa, and M. Zajac. A subversion-resistant SNARK. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 3–33. Springer, Heidelberg, Dec. 2017. 4
2. S. Atapoor and K. Baghery. Simulation extractability in Groth's zk-SNARK. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology - ESORICS 2019 International Workshops, DPM 2019 and CBT 2019*, volume 11737 of *LNCS*, pages 336–354. Springer, 2019. 2, 3, 4
3. K. Baghery. On the efficiency of privacy-preserving smart contract systems. In *AFRICACRYPT 19*, LNCS, pages 118–136. Springer, Heidelberg, 2019. 2, 4
4. K. Baghery. Subversion-resistant simulation (knowledge) sound NIZKs. In *17th IMA International Conference on Cryptography and Coding*, LNCS, pages 42–63. Springer, Heidelberg, Dec. 2019. 4
5. K. Baghery, A. González, Z. Pindado, and C. Ràfols. Signatures of knowledge for boolean circuits under standard assumptions. In *AFRICACRYPT 20*, LNCS, pages 24–44. Springer, Heidelberg, 2020. 4
6. K. Baghery, M. Kohlweiss, J. Siim, and M. Volkhov. Another look at extraction and randomization of Groth's zk-SNARK. Cryptology ePrint Archive, Report 2020/811, 2020. `https://eprint.iacr.org/2020/811`. 2, 4
7. E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014. 1, 2
8. D. Bernhard, G. Fuchsbauer, and E. Ghadafi. Efficient signatures of knowledge and DAA in the standard model. In M. J. Jacobson Jr., M. E. Locasto, P. Mohassel, and R. Safavi-Naini, editors, *ACNS 13*, volume 7954 of *LNCS*, pages 518–533. Springer, Heidelberg, June 2013. 4
9. D. Boneh and X. Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, Apr. 2008. 3, 6

10. J. Bonneau, I. Meckler, V. Rao, and E. Shapiro. Coda: Decentralized cryptocurrency at scale. Cryptology ePrint Archive, Report 2020/352, 2020. https://eprint.iacr.org/2020/352. 2
11. S. Bowe and A. Gabizon. Making groth's zk-SNARK simulation extractable in the random oracle model. Cryptology ePrint Archive, Report 2018/187, 2018. https://eprint.iacr.org/2018/187. 2, 3, 6, 8, 10, 13
12. S. Bowe, A. Gabizon, and I. Miers. Scalable multi-party computation for zk-SNARK parameters in the random beacon model. Cryptology ePrint Archive, Report 2017/1050, 2017. http://eprint.iacr.org/2017/1050. 7
13. M. Chase and A. Lysyanskaya. On signatures of knowledge. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 78–96. Springer, Heidelberg, Aug. 2006. 2, 4
14. G. Fuchsbauer. Subversion-zero-knowledge SNARKs. In M. Abdalla and R. Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 315–347. Springer, Heidelberg, Mar. 2018. 4
15. R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct NIZKs without PCPs. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Heidelberg, May 2013. 1
16. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, Dec. 2006. 2, 4
17. J. Groth. On the size of pairing-based non-interactive arguments. In M. Fischlin and J.-S. Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016. 1, 2, 3, 5, 6, 10, 13
18. J. Groth and M. Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 581–612. Springer, Heidelberg, Aug. 2017. 2, 3, 4, 5
19. T. Kerber, A. Kiayias, M. Kohlweiss, and V. Zikas. Ouroboros crypsinous: Privacy-preserving proof-of-stake. In *2019 IEEE Symposium on Security and Privacy*, pages 157–174. IEEE Computer Society Press, 2019. 2
20. A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE Symposium on Security and Privacy*, pages 839–858. IEEE Computer Society Press, May 2016. 1, 2
21. H. Lipmaa. Simulation-extractable SNARKs revisited. Cryptology ePrint Archive, Report 2019/612, 2019. http://eprint.iacr.org/2019/612. 2, 3
22. B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013. 1