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 NIZK arguments, zk-SNARKs are the most efficient constructions in terms of proof size and verification which are two critical criteria for large scale applications. Currently, Groth's construction, Groth16, from Eurocrypt'16 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 (SE) version of Groth16 proposed by Bowe and Gabizon that has the most efficient prover and crs size among the candidates, although it adds Random Oracle (RO) to the original construction. We present a new version which requires 4 parings in the verification, instead of 5. We also get rid of the RO at the cost of a collision resistant hash function and a single new element in the crs. Our construction is proven in the *generic group model* and seems to result in the most efficient SE variant of Groth16 in most dimensions.

Keywords: 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 and very efficient verification for NP complete languages [6]. Zk-SNARKs have had a tremendous impact in practice and they have found numerous applications, including verifiable computation systems [11], privacy-preserving cryptocurrencies [3] and smart contracts [9] and private proof-of-stake

protocols [8] are few of known applications that use zk-SNARKs to prove different statements while guaranteeing privacy of the users. Because of their practical importance, particularly in large-scale applications like blockchains, even minimal savings (in proof size or verification cost) are considered to be relevant. In practice, the zk-SNARK is used to prove the correctness of some computations without leaking any information about the secret inputs that are used to complete it. To do so, the computation should be encoded to one of the NP characterizations which currently Quadratic Arithmetic Program (QAP) is the most popular one. The basic idea is that the correctness of all the computations of the circuit is expressed as a divisibility relation among certain polynomials which define the program. Then the characterization can be compiled into a zk-SNARK where the prover gives a proof of knowledge of a witness $\vec{a}$ for which the divisibility relation holds for the polynomials which define the QAP combined with the input $\vec{a}$. The succinctness of the argument comes precisely from the fact that the correctness of all the gates is aggregated into just one relation, and that this relation of polynomials is proven in one secret point.

In 2016, Groth [6] introduced the most efficient zk-SNARK in the Generic Group Model (GGM) for QAPs, Groth16, which is still the state-of-the-art. Its proof is 3 group elements and its verification is dominated by 3 pairings. The proof in Groth16 is malleable. Generating non-malleable proofs is a necessary requirement in building various cryptographic schemes, including *universally composable* protocols [9, 8], cryptocurrencies (e.g. Zcash) [3], signature-of-knowledge schemes [7], etc.

Therefore, in practice, it is important to have a stronger notion of security, namely, 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 queries. For this reason, in Crypto 2017, Groth and Maller [7] 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. Bowe and Gabizon [4] proposed a less efficient construction (5 group elements vs 3) based on Groth16 which adds a Random Oracle (RO) to it but with almost no overhead in crs size or cost for the prover. Recently, Lipmaa [10] proposed several constructions, including the most efficient QAP-based SE zk-SNARK in terms of proof size and with the same verification complexity as [7, 4], but less efficient in terms of crs size and prover time compared to [4]. In [1], 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 [1].

1.1 Our Contributions

The core of our contribution is revisiting two SE variants of Groth16, presented in [4] and [1], to get the best of both constructions. Our focus is manly on Bowe and Gabizon's variation [4] which has the most efficient prover and the shortest crs among all SE zk-SNARKs [7, 4, 10, 1], while requires a RO. To achieve 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 NIZK PoK of DLOG of $[\delta']_2$ w.r.t $[\delta]_2$.

We propose a new SE variant of Groth16 based on Bowe and Gabizon's scheme [4] without ROs. Our variant uses some sophisticated modification of Boneh-Boyen signatures to prove knowledge of the DLOG of δ' and relies only on the collision-resistant properties of a hash function, apart from the GGM. In terms of efficiency, in comparison with [4], our construction requires 1 paring less in the verification, while retaining all the other properties of their construction. More specifically, the most interesting features of Bowe and Gabizon's scheme are that the crs size and the prover complexity that are almost the same as Groth16 (except for a few exponentiations). Our construction inherits these nice features and avoids using ROs, in the cost of a single new element in the crs which is negligible [3]). In comparison with [1], our variant does not have an overhead in proof generation and crs size and it also comes with smaller overhead in proof size. [4]

Tab. 1 presents a comparison of our proposed variant of Groth16 with several related constructions for a particular instance of arithmetic circuit satisfiability. Our construction gets rid of the RO in cost of adding one element to the crs.

2 Preliminaries

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.

Security for zk-SNARKs. We use the definitions of NIZK arguments from [6, 7]. Our argument is perfectly complete (honest arguments will be accepted with

[3] In the full version, we show that using a RO we can set $\gamma = 0$ and do not need to add any new element.

[4] Our changes add only one element to the crs of Groth16 and since the original version is proven to achieve subversion ZK (ZK without trusting a third party) [5], our variant also can be proven to achieve Sub-ZK using the technique proposed in [2].

Table 1. A comparison of our proposed variant of Groth16 along with 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). In the case of crs size and Prover's computation constants are omitted. In [7], n Mul gates and m wires translate to $2n$ squaring gates and $2m$ wires. In [1], $n' \approx n + 52.000$ and $m' \approx m + 52.000$. $\mathbb{G}_1$ and $\mathbb{G}_2$: group elements, E_i: exponentiation in group $\mathbb{G}_i$, M_i: multiplication in group $\mathbb{G}_i$, P: pairings, ROM: Random Oracle Model, CRH: Collision Resistant Hash.

SNARK	Security	Model	crs	Prover	Proof	Verifier
Groth [6]	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 [7]	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 [4]	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 [1]	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 E_1 4 P
Lipmaa [10]	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 E_1 5 P
This paper	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 4 P

probability 1), perfect zero-knowledge (simulated proofs have the same distribution as honest proofs) and SE (even after seeing v simulated proofs, from any accepting proof output by the adversary it is possible to extract a valid witness).

3 Simulation Extractability without Random Oracles

In this section, we propose a variation of Groth16 inspired on its Bowe and Gabizon [4] SE version. To achieve so, their prover replaces all the computations which depend on δ given in the crs by some δ' of its choice, that it must give as part of the proof, together with a proof of knowledge of the DLOG of δ' w.r.t to δ, which given some element $[Y]_1 = H([A]_1||[B]_2||[C]_1||[\delta']_2)$, consists of $[\pi]_1$ such that $e([Y]_1, [\delta']_2) = e([\pi]_1, [\delta]_2)$. In their analysis, H is an RO and their proof requires 2 pairings for verification. Our contribution is to give an alternative argument of knowledge for the DLOG, with a novel use of Boneh-Boyen signatures along with a proof in the GGM.

Scheme definition. In Fig. 1, we present our version of Groth16 and we explain in the following how we avoid the use of RO.

Avoiding RO. Our proof uses the collision resistance property of the hash function and the GGM. 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 $\delta' + \delta m = \zeta\delta$.

Setup, $\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$**:** Pick $x, \alpha, \beta, \delta \leftarrow \mathbb{Z}_p^*$, $H \leftarrow \mathcal{H}$ and returns crs, where

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

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

1. Select a random element $\zeta \leftarrow \mathbb{Z}_p^*$, and set $[\delta']_2 := \zeta[\delta]_2$
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 $[h(x)t(x)/\delta']_1 \leftarrow (1/\zeta)(\sum_{i=0}^{n-2} h_i [x^i t(x)/\delta]_1)$,
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 $[C]_1 \leftarrow r_b[A]_1 + r_a \left(\sum_{j=0}^m a_j [v_j(x)]_1 + [\beta]_1\right) + (1/\zeta) \sum_{j=l+1}^m a_j ([(u_j(x)\beta + v_j(x)\alpha + w_j(x))/\delta]_1) + [h(x)t(x)/\delta']_1$,
8. Sets $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$,
9. Compute $[D]_1 = \frac{m}{\zeta+m}[\frac{t(x)}{\delta}]_1 + \frac{1}{\zeta+m}[\frac{\gamma t(x)}{\delta}]_1 = [\frac{(m+\gamma)t(x)}{\delta'+m\delta}]_1$
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 $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$ check 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 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 $\vec{\mathsf{ts}} := (x, \alpha, \beta, \delta)$ act as follows,

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

Fig. 1. The proposed variation of Groth16 for $\mathbf{R}$. $\mathcal{H}$ 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. Differences with Groth16 are highlighted. Alternatively, one can describe Groth16 as corresponding to $\zeta = 1, \gamma = 0$ and where the proof consists only of $[A, C]_1, [B]_2$.

Security. We prove security of new scheme (Fig. 1) in Theorem 1.

Theorem 1 (Completeness, ZK, SE). *The variation of Groth16 described in Fig. 1, 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 by reduction (in the GGM) to the knowledge soundness of Groth16. The reduction works in two steps (similarly to [4], 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 Groth16.

Proof of Step 1) Suppose $\mathcal{A}$ has made queries $\vec{x}_1, \cdots, \vec{x}_v$ 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,

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

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 $\mathcal{A}$ has produced the elements (A, B, C, D, δ') such that $A \cdot B \equiv C \cdot \delta' + \left(\sum_{j=0}^{l} a_j (u_j(x)\beta + v_j(x)\alpha + w_j(x))\right) + \alpha\beta$, 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' be 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 Q_2' for B, δ') and we can extract the coefficients of this linear combination.

First, we prove that $\mathcal{A}$ has knowledge of the DLOG 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,

$$t(x)(m + \gamma) = (\textstyle\sum_{q \in Q_1'} k_{D,q}\ q)(\delta' + m\delta), \tag{1}$$

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 Eq. (1), we obtain

$$t(x)\gamma = k_D \tfrac{\gamma t(x)}{\delta}(\delta' + m\delta) + \textstyle\sum_{j=1}^{v} k_{D,j} \frac{\gamma t(x)}{\delta_j + m_j\delta}(\delta' + m\delta). \tag{2}$$

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

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

$$\Leftrightarrow \prod_{i=0}^{v}(\delta_i + m_i\delta) = (\delta' + m\delta)\left(\sum_{j=0}^{v} k_{D,j} \prod_{i=0,i\neq j}^{v}(\delta_i + m_i\delta)\right). \quad (3)$$

It follows that the term $\delta' + m\delta$ must divide the left side of Eq. (3). 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. (3) by $(\delta_{j^*} + m_{j^*}\delta)$, we come to the following expression:

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

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) $\delta' + \delta m = k\delta$, in which case we can extract the DLOG of δ' as $k - m$ as wanted, or (2) $\delta' + \delta m = k(\delta_{j^*} + m_{j^*}\delta)$, in which case, from Eq. (1) 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 [6], 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 [4] 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 $\frac{t(x)(m_j+\gamma)}{\delta_j + m_j\delta}$ appears in the expression of A generated from Q_1' with the corresponding coefficient different from 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'.

If D_j appears in C, then the term $C\delta'$ 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 it can only appear in AB. Since $\delta' \in Q_2'$, then A would contain D_j, which we ruled out previously. □

4 Conclusion

Over the last few years, various zk-SNARKs have been proposed that achieve simulation extractability [7, 4, 10, 1], which is a requirement for zk-SNARKs

to generate non-malleable proofs. In this paper, we revised the SE variation of Groth16 proposed in [4] and presented a new one. Our construction requires 4 pairings in verification, instead of 5 in [4], and also avoids ROs in exchange for using a collision resistant hash function. It has a more efficient prover, crs size, and proof size in comparison with [1], that has also 4 pairings in verification.

Acknowledgements. The research leading to this article was partially supported by Project RTI2018-102112-B-I00 (AEI/FEDER, UE), Defense Advanced Research Projects Agency(DARPA) under Contract No. HR001120C0085, and by Cyber Security Research Flanders with reference number VR20192203.

References

1. 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, 7, 8
2. 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. 3
3. 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
4. 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, 4, 6, 7, 8
5. 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. 3
6. 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, 4, 7
7. 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, 7
8. 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
9. 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
10. H. Lipmaa. Simulation-extractable SNARKs revisited. Cryptology ePrint Archive, Report 2019/612, 2019. `http://eprint.iacr.org/2019/612`. 2, 3, 4, 7
11. 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