Simulation Extractability in Groth's zk-SNARK

Shahla Atapoor
Institute of Computer Science
University of Tartu
Tartu, Estonia
shahla.atapoor@ut.ee

Karim Baghery
Institute of Computer Science
University of Tartu
Tartu, Estonia
karim.baghery@ut.ee

***Abstract*—A Simulation Extractable (SE) zk-SNARK enables a prover to prove that she knows a witness for an instance in a way that the proof: (1) is succinct and can be verified very efficiently; (2) does not leak information about the witness; (3) is simulation-extractable -an adversary cannot come out with a new valid proof unless it knows a witness, even if it has already seen arbitrary number of simulated proofs. Non-malleable succinct proofs and very efficient verification make SE zk-SNARKs an elegant tool in various privacy-preserving applications such as cryptocurrencies, smart contracts and etc. In Eurocrypt 2016, Groth proposed the most efficient pairing-based zk-SNARK in the CRS model, but its proof is vulnerable to malleability attacks. In this paper, we show that one can efficiently achieve simulation extractability in Groth's zk-SNARK by some changes in the underlying language using an OR construction. Analysis show that in practical cases overload has minimal effects on the efficiency of original scheme which currently is the most efficient zk-SNARK. In new construction, proof size will be extended by one element from $\mathbb{G}_1$, one element from $\mathbb{G}_2$ plus a bit string, that totally will be still less than 200 bytes for 128-bit security. Its verification is dominated with 4 parings which is the most efficient verification among current SE zk-SNARKs.**

***Index Terms*—Zero-knowledge proofs, zk-SNARKs, simulation extractability, CRS model**

I. Introduction

Non-Interactive Zero-Knowledge (NIZK) proofs are one of the central design tools in cryptographically secure systems, allowing one to verify the veracity of statements without leaking extra information. Technically speaking, a NIZK allows a prover to prove that, for a public statement x she knows a witness w which hold in a relation $\mathcal{R}$, $(x, w) \in \mathcal{R}$, without leaking any information about her witness w. In the Common Reference String (CRS) model [1], a NIZK is a three-party protocol that works as the following. First, there exists a trusted party K (a.k.a. CRS generator) who takes security parameter λ as an input and generates CRS elements $\mathsf{crs} := (\mathsf{crs}_\mathsf{P}, \mathsf{crs}_\mathsf{V})$ which later will be used by prover and verifier for proof generation and proof verification, respectively. Then the prover P gets crs_P, the statement x and her witness w and generates a proof π, attesting that for the statement x, I know a witness w s.t. $(x, w) \in \mathcal{R}$. Finally, a verifier V takes crs_V, the statment x and the proof π and returns either accept (if proof is valid) or reject (if verification failed). If V does not need any secret information to verify the proof π, the proof is called *publicly verifiable* that can be verified by many public verifiers (e.g. by nodes of a distributed network).

Generally, a NIZK proof system guarantees three essential properties known as *completeness*, *soundness* and *zero-knowledge*. The property *completeness* guarantees that a honest prover always convinces a honest verifier. The *soundness* ensures that a malicious prover cannot convince the honest verifier except with negligible probability. Zero-knowledge property assures that the proof generated by prover does not leak any information about the witness w of prover.

During last few years, a very efficient family of NIZK proof systems are developed which are known as zero-knowledge Succinct Non-interactive Argument of Knowledge (zk-SNARK) [2]–[7]. A zk-SNARK has *succinct* proof size that allows a computationally weak verifier to efficiently verify the proof. Differently from a standard NIZK, a zk-SNARK guarantees *knowledge-soundness* that is a stronger notion in comparison with standard soundness. Knowledge-soundness guarantees that if an adversarial prover manages to come out with an acceptable proof, there exists an efficient extractor which given some secret information can efficiently extract the witness from the proof. Knowledge-soundness of zk-SNARKs is achieved under knowledge assumptions [8] that allow an extractor to extract witness from a succinct proof in a non-black-box manner[1]. Impossibility result of Gentry and Wichs [9] confirms that such kind of extraction should be based on non-falsifiable (e.g. knowledge assumptions [8]) assumptions. By the date, the most efficient zk-SNARK is proposed by Groth [6] in Eurocrypt 2016, that is constructed for Quadratic Arithmetic Programs (QAPs) and works in a biliner group. The proof in Groth's zk-SNARK consists of two elements in $\mathbb{G}_1$ and one element in $\mathbb{G}_2$, and in order to verify the proof, a verifier V needs to check one equation that contains 3 parings and m_0 exponentiations [6], where m_0 is the length of inputs (statements) to the circuit.

In practice, however knowledge-soundness is an amplified notion in comparison with standard soundness, but still a knowledge-sound proof is vulnerable to the man-in-the-middle attacks [2]. Due to this fact, zk-SNARKs that only guarantee knowledge-soundness cannot be deployed in many of prac-

[1]In non-black-box extraction, extractor $\mathcal{E}_\mathcal{A}$ needs to get full access to the source code and random coins of adversary $\mathcal{A}$ to be able to extract the witness.

[2]For instance, in the verification equations that have paring structure such as $\mathfrak{a} \bullet \mathfrak{b} = \cdots$, where $\mathfrak{a}$ and $\mathfrak{b}$ are proof elements from $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime orders, one can see that such verification equation will be satisfied also for new proof elements such as $\mathfrak{a}' = r\mathfrak{a}$ and $\mathfrak{b}' = \frac{1}{r}\mathfrak{b}$, for arbitrary $r \leftarrow \mathbb{Z}_p$.

tical applications straightforwardly [10]–[13]. For instance, privacy-preserving crypocurrencies such as Zerocash that uses zk-SNARKs as a subroutine, takes extra steps to prevent malleability attacks in the proofs for *pour* transactions [10]. Similarly, privacy-preserving smart contract systems [11], [12] show that knowledge-soundness of zk-SNARKs is not enough for their systems. *Simulation-knowledge soundness* which also is known as *simulation extractability* [14], is an amplified version of knowledge-soundness that is proposed to provide non-malleability in NIZK proofs. A Simulation-Extractable (SE) zk-SNARK guarantees the proof is *succinct*, *zero-knowledge* (does not leak any information about w) and *simulation-extractable*, meaning that an adversarial prover is unable to come out with a new proof unless he knows a witness, even if he has already seen arbitrary number of simulated proofs. Intuitively, simulation extractability implies that the proofs are non-malleable and even if an adversary sees arbitrary number of simulated proofs, he will not be able to generate a new acceptable proof, if he could, he must know the witness.

In Crypto 2017, Groth and Maller proposed the first SE zk-SNARK in the CRS model that allows to generate non-malleable proofs [7]. They also proved that a SE zk-SNARK requires at least two verification equations. Their scheme is constructed in the bilinear groups for Square Arithmetic Programs (SAPs) and its proof consists of two elements in $\mathbb{G}_1$ and one element in $\mathbb{G}_2$. In order to verify a proof, their verifier needs to check two equations that totally require 5 pairings and m_0 exponentiations, m_0 is the length of statement [7]. One may notice that Groth-Maller scheme is optimal in the number of verification equations, but to guarantee non-malleability in proofs, their scheme removes one of the bilinear group generators from the CRS, which creates some different challenges in practice (e.g. in CRS generation by multi-party computation protocols, or in achieving subversion security [15]). Above all, Groth-Maller scheme is constructed for arithmetic circuits with squaring gates that requires larger number of gates, as each multiplication (MUL) gate requires two squaring gates ($a \times b = \frac{(a+b)^2-(a-b)^2}{4}$). Finally, implementations show that for a particular circuit, Groth's zk-SNARK [6] considerably has better efficiency than Groth-Maller scheme [7], but Gorth's scheme does not achieve simulation extractability (its proof is malleable). A short performance comparison of both schemes on a Rank-1 Constraint System (R1CS) instance with 10^6 constraints and 10^6 variables, where 10 are input variables is shown in Tab. I.

Table I

A short performance comparison of zk-SNARKs proposed by Groth and Maller [7] (reffered as GM), and Groth [6] for arithmetic circuit satisfiability with an R1CS instance with 10^6 constraints and 10^6 variables, where 10 are input variables. Since [7] uses squaring gates, so n MUL gates translate to $2n$ squaring gates. Implementations are done on a PC with 3.40 GHz Intel Core i7-4770 CPU, in single-threaded mode using the BN128 curve.

zk-SNARK	CRS Gen. CRS Size	Pro. Time Proof Size	Ver. Time Ver. Comp.	Security
GM [7]	100.4 sec 385 MB	116.4 sec $2\mathbb{G}_1 + 1\mathbb{G}_2$	2.3 msec 5 paring	Simulation Extractability
Groth [6]	72.5 sec 201 MB	84.0 sec $2\mathbb{G}_1 + 1\mathbb{G}_2$	1.3 msec 3 paring	Knowledge Soundness

Problem statement. By reminding that currently zk-SNARK of Groth [6] has the best efficiency but only achieves knowledge-soundness, and the fact that Groth-Maller zk-SNARK [7] ensures simulation extractability but with less efficiency and only one group generator in the CRS, a research question can be raised as if we can achieve simulation extractability in Groth's zk-SNARK efficiency? Such that, new scheme will 1) work for QAPs 2) have both generators of bilinear groups in the CRS 3) have a comparable or even better efficiency than Groth-Maller zk-SNARK.

Our Contribution. In this paper, we address the questions discussed above and propose a variation of Groth's zk-SNARK that can achieve simulation extractability with minimal efficiency loss in practical cases. To this end, we use the known OR technique and define a new language $\mathcal{L}'$ based on the language $\mathcal{L}$ in Groth's zk-SNARK that is inspired by the works of De Santis et al. [16] and Kosba et al. [17].

Defining new language implies some changes in algorithms of the original scheme. Our evaluations show that for a particular practical instantiation, new changes have minimal affect on efficiency of the original scheme which currently is the state-of-the-art zk-SNARK. Strictly speaking, the verification of new scheme has two equations as the optimal case, and only adds 1 pairing to the verification of Groth's scheme. As a result, totally verification (Ver. Comp.) of new scheme requires 4 pairings which is less than 5 parings in the current state-of-the-art simulation-extractable zk-SNARK proposed by Groth and Maller [7]. So, by considering Tab. I, the verification (Ver. Time) would take approximately 1.8 milliseconds. In the proposed scheme, the proof size will be extended by one element from $\mathbb{G}_1$, one element from $\mathbb{G}_2$ plus a 256-bit string, that totally (Proof Size) will be 3 elements from $\mathbb{G}_1$, 2 elements from $\mathbb{G}_2$ and one 256-bit string, which for the considered instance in Tab. I still it would be less than 200 bytes for 128-bit security. The prover will require to give a proof for a new circuit that has approximately 50×10^3 gates more than before, which in practice the overload is negligible. For instance Zerocash uses zk-SNARKs to give a proof in the *pour* transactions that use an arithmetic circuit with around 2×10^6 MUL gates[3]. As for a circuit with size c, the prover does asymptotically $O(c)$ cryptographic and $O(c \log^2 c)$ non-cryptographic computations, so for circuits with 10^6 gates, the overload would be around 5-6%, where by considering reports in Tab. I, the proof generation (Pro. Time) would take about 90 seconds. Similarly, for the same instance in Tab. I, the CRS size in new scheme will increase around 5-6% which will result around 215 MB, considerably smaller than CRS size of Groth-Maller zk-SNARK [7] that chases the same goal.

Discussion and Related Works. Among different NIZK arguments, zk-SNARKs are the most practically-interested ones, because of their succinct proofs and very efficient verifications. But as majority of them guarantee knowledge-soundness by

[3]Infact already their circuit for *pour* transactions had around 4×10^6 gates, but recently they optimized the system and reduced the number of gates, but still it has around 40 times 50×10^3 gates.

default, that is vulnerable to the man-in-the-middle attacks, so in practical systems they are supported by some other cryptographic primitives [10]–[13]. In constructing large cryptographic systems, this can make some challenges for non-expert users. Due to this fact, recently constructing efficient SE zk-SNARKs, that by default can guarantee non-malleability of proofs, has become an active research topic [7], [18], [19]. In 2018, Bowe and Ariel [18] also proposed a variation of Groth's zk-SNARK that aimed to achieve simulation extractability but in the random oracle model. However, recently Kim et al. [19] showed that Bowe -Ariel scheme still is vulnerable to the malleability attack. As Groth's zk-SNARK is constructed and proven in the CRS model, so we are interested to achieve simulation extractability in the same model. Currently, we are aware of only Groth-Maller SE zk-SNARK [7] in the CRS model, which later we will compare it with the proposed construction with more details.

The rest of paper is organized as follows; Sec. II introduces notations and preliminaries. A simulation-extractable version of Groth's zk-SNARK is presented in Sec. III. In Sec. IV, we discuss about instantiation and efficiency of the proposed construction. Finally we conclude the paper in Sec. V.

II. Preliminaries

Let PPT denote probabilistic polynomial-time, and NUPPT denote non-uniform PPT. Let $\lambda \in \mathbb{N}$ be the information-theoretic security parameter, say $\lambda = 128$. All adversaries will be stateful. 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 $\mathcal{A}$, and let $r \leftarrow \mathsf{RND}(\mathcal{A})$ denote sampling of a randomizer r of sufficient length for $\mathcal{A}$'s needs. By $y \leftarrow \mathcal{A}(x;r)$ we denote the fact that $\mathcal{A}$, given an input x and a randomizer r, outputs y. 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 "$y \leftarrow \mathcal{A}(x;r)$, $y' \leftarrow \mathcal{E}_{\mathcal{A}}(x;r)$". We denote by negl an arbitrary negligible function. For distributions A and B, $A \approx_c B$ means that they are computationally indistinguishable.

In pairing-based groups, we use additive notation together with the bracket notation, i.e., in group $\mathbb{G}_\mu$, $[a]_\mu = a\,[1]_\mu$, where $[1]_\mu$ is a fixed generator of $\mathbb{G}_\mu$. A *bilinear group generator* $\mathsf{BGgen}(1^\lambda)$ returns $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, [1]_1, [1]_2)$, where p (a large prime) is the order of cyclic abelian groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$. Finally, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficient non-degenerate bilinear pairing, s.t. $\hat{e}([a]_1, [b]_2) = [ab]_T$. Denote $[a]_1 \bullet [b]_2 = \hat{e}([a]_1, [b]_2)$.

Next we review Quadratic Arithmetic Programs (QAPs) that defines NP-complete language specified by a quadratic equation over polynomials and have reduction from the language Circuit-SAT [6], [20].

Quadratic Arithmetic Programs. QAP was introduced by Gennaro *et al.* [20] as a language where for an input x and witness w, $(x, w) \in \mathcal{R}$ can be verified by using a parallel quadratic check. Consequently, any efficient simulation-extractable zk-SNARK for QAP results in an efficient simulation-extractable zk-SNARK for Circuit-SAT.

An QAP instance $\mathcal{Q}_p$ is specified by the so defined $(\mathbb{Z}_p, m_0, \{u_j, v_j, w_j\}_{j=0}^m)$. This instance defines the following relation, where we assume that $A_0 = 1$:

$$\mathcal{R} = \left\{ \begin{array}{r} (x,w)\colon x = (A_1, \ldots, A_{m_0})^\top \wedge w = (A_{m_0+1}, \ldots, A_m)^\top \wedge \\ \left(\sum_{j=0}^m A_j u_j(X)\right)\left(\sum_{j=0}^m A_j v_j(X)\right) \equiv \\ \sum_{j=0}^m A_j w_j(X) \pmod{\ell(X)} \end{array} \right\}$$

Alternatively, $(x, w) \in \mathcal{R}$ if there exists a (degree $\leq n-2$) polynomial $h(X)$, s.t. $\left(\sum_{j=0}^m A_j u_j(X)\right)\left(\sum_{j=0}^m A_j v_j(X)\right) - \sum_{j=0}^m A_j w_j(X) = h(X)\ell(X)$, where $\ell(X) = \prod_{i=1}^{\mathsf{n}}(X - \omega^{i-1})$ is a polynomial related to Lagrange interpolation, and ω is an n-th primitive root of unity modulo p. Roughly speaking, the goal of the prover of a zk-SNARK for QAP [20] is to prove that for public $(A_1, \ldots, A_{m_0})$ and $A_0 = 1$, she knows $(A_{m_0+1}, \ldots, A_m)$ and a degree $\leq n-2$ polynomial $h(X)$, such that above equation holds.

A. Definitions

We use the definitions of NIZK arguments from [6], [7], [14], [17]. Let $\mathbf{R}$ be a relation generator, such that $\mathbf{R}(1^\lambda)$ returns a polynomial-time decidable binary relation $\mathcal{R} = \{(x, w)\}$. Here, x is the statement and w is the witness. Security parameter λ can be deduced from the description of $\mathcal{R}$. The relation generator also outputs auxiliary information $\mathsf{z}_\mathcal{R}$ that will be given to the honest parties and the adversary. As in [6], [15], $\mathsf{z}_\mathcal{R}$ is the value returned by $\mathsf{BGgen}(1^\lambda)$, so $\mathsf{z}_\mathcal{R}$ is given as an input to the honest parties; if needed, one can include an additional auxiliary input to the adversary. Let $\mathcal{L}_\mathcal{R} = \{x : \exists w, (x, w) \in \mathcal{R}\}$ be an NP-language. A *NIZK argument system* Ψ for $\mathbf{R}$ consists a tuple of PPT algorithms $(\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{S})$, such that:

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

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

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

Simulator: S is a PPT algorithm that, given $(\mathcal{R}, \mathsf{z}_\mathcal{R}, \mathsf{crs}, \mathsf{ts}, x)$, outputs an argument π.

By definition, a zk-SNARK system is required to be complete, (computationally) knowledge-sound, (statistically) ZK, and succinct as in the following definitions.

Definition 1 (Perfect Completeness [6])**.** *A non-interactive argument Ψ is* perfectly complete for $\mathbf{R}$*, if for all λ, all $(\mathcal{R}, \mathsf{z}_\mathcal{R}) \in \mathbf{Im}(\mathbf{R}(1^\lambda))$, and $(x, w) \in \mathcal{R}$,*

$$\Pr\left[\begin{array}{l}(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}, \mathsf{z}_\mathcal{R}),\ \pi \leftarrow \mathsf{P}(\mathcal{R}, \mathsf{z}_\mathcal{R}, \mathsf{crs}_\mathsf{P}, x, w) : \\ \mathsf{V}(\mathcal{R}, \mathsf{z}_\mathcal{R}, \mathsf{crs}_\mathsf{V}, x, \pi) = 1\end{array}\right] = 1\ .$$

Intuitively, it states an honest prover P always convinces an honest verifier V.

Definition 2 (Computationally Knowledge-Soundness [6])**.** *A non-interactive argument Ψ is computationally (adaptively)* knowledge-sound for $\mathbf{R}$*, if for every NUPPT $\mathcal{A}$, there exists a NUPPT extractor $\mathcal{E}_{\mathcal{A}}$, s.t. for all λ,*

$$\Pr\begin{bmatrix}(\mathsf{crs}\,\|\,\mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}, \mathsf{z}_{\mathcal{R}}), r \leftarrow \mathsf{RND}(\mathcal{A}),\\ ((x,\pi)\,\|\,w) \leftarrow (\mathcal{A}\,\|\,\mathcal{E}_{\mathcal{A}})(\mathcal{R}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}; r) :\\ (x,w) \notin \mathcal{R} \wedge \mathsf{V}(\mathcal{R}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}_{\mathsf{V}}, x, \pi) = 1\end{bmatrix} \approx_{\lambda} 0 \ .$$

Here, $\mathsf{z}_{\mathcal{R}}$ can be seen as a common auxiliary input to $\mathcal{A}$ and $\mathcal{E}_{\mathcal{A}}$ that is generated by using a benign [21] relation generator. Intuitively, the definition states that if an adversary can convince the verifier, she *knows* the witness. A knowledge-sound Ψ also is called an *argument of knowledge*.

Definition 3 (Statistically Zero-Knowledge (ZK) [6])**.** *A non-interactive argument Ψ is* statistically ZK for $\mathbf{R}$*, if for all λ, all $(\mathcal{R}, \mathsf{z}_{\mathcal{R}}) \in \mathbf{Im}(\mathbf{R}(1^{\lambda}))$, and for all NUPPT $\mathcal{A}$, $\varepsilon_0^{unb} \approx_{\lambda} \varepsilon_1^{unb}$, where*

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

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

Intuitively, a non-interactive argument Ψ is zero-knowledge if it does not leak extra information beside the truth of the statement.

Definition 4 (Succinctness [7])**.** *A non-interactive argument Ψ is* succinct *if the proof size is polynominal in λ and the verifier's computation time is polynominal in security parameter λ and the size of instance x.*

In the rest, we recall the definition of (non-black-box) simulation extractability that we aim to achieve in the proposed variation of Groth's zk-SNARK.

Definition 5 ((Non-Black-Box) Simulation Extractability [7])**.** *A non-interactive argument Ψ is* (non-black-box) simulation-extractable for $\mathbf{R}$*, if for any NUPPT $\mathcal{A}$, there exists a NUPPT extractor $\mathcal{E}_{\mathcal{A}}$ s.t. for all λ,*

$$\Pr\begin{bmatrix}(\mathsf{crs}\,\|\,\mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}, \mathsf{z}_{\mathcal{R}}), r \leftarrow_r \mathsf{RND}(\mathcal{A}),\\ ((x,\pi)\,\|\,w) \leftarrow (\mathcal{A}^{\mathsf{O}(\cdot)}\,\|\,\mathcal{E}_{\mathcal{A}})(\mathcal{R}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}; r) :\\ (x,\pi) \notin Q \wedge (x,w) \notin \mathcal{R} \wedge \mathsf{V}(\mathcal{R}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}_{\mathsf{V}}, x, \pi) = 1\end{bmatrix} \approx_{\lambda} 0 \ .$$

Here, Q is the set of simulated statement-proof pairs generated by the adversary's queries to O. It is worth to mention that *(non-black-box) simulation extractability* implies *knowledge-soundness* (given in Def. 2), as the earlier is a strong notion of the later which additionally the adversary is allowed to send query to the proof simulation oracle. Note that in Def. 5 the extractor is non-black-box and to be able to extract the witness, it should have access to the source code and the random coins of $\mathcal{A}$.

III. A Variation of Groth's zk-SNARK

As briefly discussed in the introduction, Groth's zk-SNARK [6] guarantees knowledge-soundness (defined in Def. 2) which is weaker than simulation extractability. Technically speaking, knowledge-sound proofs are not secure against the man-in-the-middle attacks. In this section we present a variation of Groth's zk-SNARK which can achieve (non-black-box) simulation extractability, defined in Def. 5, that can guarantee non-malleability of the proofs.

A. New Construction

In new construction, we use an OR technique [16], [17] which adds some efficient computations for all algorithms of the argument but keeps the internal computations of prover and verifier as the original one that considerably are optimized for the QAP relation. More precisely, initially we define a new language $\mathcal{L}'$ based on the language $\mathcal{L}$ in Groth's zk-SNARK that is embedded with commitment of a secret randomness as a key for a pseudorandom number generator. Let $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{SigVerify})$ be a one-time signature scheme and $(\mathsf{Com}, \mathsf{ComVerify})$ be a perfectly binding commitment scheme. Given the language $\mathcal{L}$ with the corresponding NP relation $\mathcal{R}_{\mathcal{L}}$, we define a new language $\mathcal{L}'$ such that $((x, \mu, pk_{\mathsf{Sign}}, \rho), (w, r, s)) \in \mathcal{R}_{\mathcal{L}'}$ iff:

$$((x,w) \in \mathcal{R}_{\mathcal{L}} \vee (\mu = f_s(pk_{\mathsf{Sign}}) \wedge \rho = \mathsf{Com}(s, r))) \,,$$

where $\{f_s : \{0,1\}^* \to \{0,1\}^{\lambda}\}_{s \in \{0,1\}^{\lambda}}$ is a pseudo-random function family.

Now, (knowledge-sound) zk-SNARK of Groth for the relation $\mathcal{R}$ constructed from PPT algorithms $(\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{S})$ can

CRS generator $\mathsf{K}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}})$**:** Sample $(\mathsf{crs}\,\|\,\mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s, r \leftarrow_u \{0,1\}^{\lambda}$; $\rho := \mathsf{Com}(s, r)$; and output $(\mathsf{crs}'\,\|\,\mathsf{ts}') := ((\mathsf{crs}, \rho)\,\|\,(\mathsf{ts}, (s, r)))$; where ts' is new simulation trapdoor.

Prover $\mathsf{P}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, w)$**:** Parse $\mathsf{crs}' := (\mathsf{crs}, \rho)$; abort if $(x,w) \notin \mathcal{R}_{\mathcal{L}}$; generate $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^{\lambda})$; sample $z_0, z_1, z_2 \leftarrow_u \{0,1\}^{\lambda}$; generate $\pi \leftarrow \mathsf{P}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, z_0, pk_{\mathsf{Sign}}, \rho), (w, z_1, z_2))$ using the prover of Groth's scheme; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, z_0, \pi))$; and return $\pi' := (z_0, \pi, pk_{\mathsf{Sign}}, \sigma)$.

Verifier $\mathsf{V}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, \pi')$**:** Parse $\mathsf{crs}' := (\mathsf{crs}, \rho)$ and $\pi' := (z_0, \pi, pk_{\mathsf{Sign}}, \sigma)$; abort if $\mathsf{SigVerify}(pk_{\mathsf{Sign}}, (x, z_0, \pi), \sigma) = 0$; call the verifier of Groth's scheme $\mathsf{V}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, z_0, pk_{\mathsf{Sign}}, \rho), \pi)$ and abort if it outputs 0.

Simulator $\mathsf{S}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', \mathsf{ts}', x)$**:** Parse $\mathsf{crs}' := (\mathsf{crs}, \rho)$ and $\mathsf{ts}' := (\mathsf{ts}, (s, r))$; generate $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^{\lambda})$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (\mathsf{ts}\,\|\,(s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; and output $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.

Figure 1. A variation of Groth's zk-SNARK.

be lifted to a simulation-extractable zk-SNARK Ψ' with PPT algorithms $(\mathsf{K}', \mathsf{P}', \mathsf{V}', \mathsf{S}')$ as described in Fig. 1.

B. Security Proofs

Here we present security proofs of the proposed scheme.

Theorem 1 (Completeness)**.** *The variation of Groth's zk-SNARK described in Sec. III-A, guarantees completeness.*

Proof. As mentioned before, in the new construction internal computations of P and V are the same as original one, except few extra efficient computations. More precisely, P needs to do the computation for a new instance that has slightly larger size (e.g. $n = n_{old}+n_{new}$, where n_{new} is the number of MUL gates added to the old circuit in result of our changes) and also output few new elements in result of singing the proof with a one-time secure signature. On the other side, extra from the original scheme, V requires to verify a one-time secure signature which can be done very efficiently.

So by considering completeness of the original scheme, and the fact that signature scheme is complete, where implies $\mathsf{SigVerify}(pk_{\mathsf{Sign}}, m, \mathsf{Sign}(m, sk_{\mathsf{Sign}}) = 1$, one can conclude that the modified construction satisfies the completeness. ■

Theorem 2 (Zero-Knowledge)**.** *The variation of Groth's zk-SNARK described in Sec. III-A, guarantees computational zero-knowledge.*

Proof. To prove the theorem we write a series of hybrid experiments which start from an experiment with the simulator and gets to an experiment that uses the real prover. We show that all experiments are two-by-two indistinguishable. Before describing the games, recall that Groth's zk-SNARK guarantees perfect zero-knowledge and simulation procedure of the modified scheme is expressed in Fig. 1. Now consider the following experiments,

$\underline{\mathsf{EXP}_1}$(simulator):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s, r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s, r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s, r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x, w)$: Abort if $(x, w) \notin \mathcal{R}_{\mathcal{L}}$; $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), \mathsf{ts}')$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $b \leftarrow \mathcal{A}^{\mathsf{O}(x,w)}(\mathsf{crs}')$

$\underline{\mathsf{EXP}_2}$(separate secret key of pseudo random function):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s', s, r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s', r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s, s', r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x, w)$: Abort if $(x, w) \notin \mathcal{R}_{\mathcal{L}}$; generate $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (\mathsf{ts} \,\|\, (s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $b \leftarrow \mathcal{A}^{\mathsf{O}(x,w)}(\mathsf{crs}')$

Lemma 1. *If the underlying commitment scheme is computationally hiding, then for two experiments* EXP_2 *and* EXP_1 *we have* $\Pr[\mathsf{EXP}_2] \approx \Pr[\mathsf{EXP}_1]$.

Proof. Computationally hiding property of a commitment scheme implies that $\mathsf{Com}(m_1, r)$ is computationally indistinguishable from $\mathsf{Com}(m_2, r)$. So this property straightforwardly results the lemma. ■

$\underline{\mathsf{EXP}_3}$(replace pseudo random function):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; s', ~~s~~, $r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s', r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (\not{s}, s', r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x, w)$: Abort if $(x, w) \notin \mathcal{R}_{\mathcal{L}}$; $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu \leftarrow_u \{0,1\}^\lambda$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (\mathsf{ts} \,\|\, (s', r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $b \leftarrow \mathcal{A}^{\mathsf{O}(x,w)}(\mathsf{crs}')$

Lemma 2. *If the underlying pseudo random function* $f_s(\cdot)$ *is secure and the underlying one-time signature scheme is unforgeable, we have* $\Pr[\mathsf{EXP}_3] \approx \Pr[\mathsf{EXP}_2]$.

Proof. By considering that the signature scheme is secure, we note that the generated pk_{Sign} is unique with overwhelming probability. Additionally, we can replace the pseudo random function $f_s(\cdot)$ with a true random function that will result EXP_4. By considering unique values of pk_{Sign} and indistinguishability of output of $f_s(\cdot)$ and truly random function, one can conclude the claim. ■

$\underline{\mathsf{EXP}_4}$(prover):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s', r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s', r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s', r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x, w)$: Abort if $(x, w) \notin \mathcal{R}_{\mathcal{L}}$; $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu \leftarrow_u \{0,1\}^\lambda$ (μ plays the role of z_0 in Fig. 1); sample $z_1, z_2 \leftarrow_u \{0,1\}^\lambda$; generate $\pi \leftarrow \mathsf{P}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (w, z_1, z_2))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $b \leftarrow \mathcal{A}^{\mathsf{O}(x,w)}(\mathsf{crs}')$

Lemma 3. *If Groth's zk-SNARK guarantees zero-knowledge, then we have* $\Pr[\mathsf{EXP}_4] \approx \Pr[\mathsf{EXP}_3]$.

Proof. The last experiment exactly models the real prover of construction in Fig. 1, and as already Groth's scheme guarantees zero-knowledge, so one can conclude that the real proof (generated by prover) in experiment EXP_4 is indistinguishable from the simulated proof (generated by simulator) in EXP_3. Intuitively this is because all new elements added to the new construction are chosen randomly and independently. ■

This results that the construction proposed in Fig. 1, guarantees computationally zero-knowledge. ■

Theorem 3 ((Non-Black-Box) Simulation Extractability)**.** *The variation of Groth's zk-SNARK, described in Sec. III-A, guarantees (non-black-box) simulation extractability.*

Proof. Similarly, we will go through a sequence of hybrid experiences that starts by the definition, and finally we show that success probability in the game is negligible; which proves the theorem. First of all, recall that Groth's scheme is proven to achieve knowledge-soundness (defined in Def. 2). Now consider the following game,

$\underline{\mathsf{EXP}_1}$(main experiment):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s, r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s, r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s, r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x)$: $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{P}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (w, (s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $(x, \pi') \leftarrow \mathcal{A}^{\mathsf{O}(x)}(\mathsf{crs}')$.
- Parse $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$; $w \leftarrow \mathcal{E}_{\mathcal{A}}(\mathsf{crs}', x, \pi, \mathsf{z}_{\mathcal{R}})$.
- Return 1 iff $((x, \pi') \notin Q) \wedge (\mathsf{V}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, \pi') = 1) \wedge ((x, w) \notin \mathcal{R}_{\mathcal{L}})$;
  where Q shows the set of statment-proof pairs generated by $\mathsf{O}(x)$.

$\underline{\mathsf{EXP}_2}$(relaxing the return checking):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s, r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s, r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s, r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x)$: $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{P}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (w, (s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $(x, \pi') \leftarrow \mathcal{A}^{\mathsf{O}(x)}(\mathsf{crs}')$.
- Parse $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$; $w \leftarrow \mathcal{E}_{\mathcal{A}}(\mathsf{crs}', x, \pi, \mathsf{z}_{\mathcal{R}})$.
- Return 1 iff $((x, \pi') \notin Q) \wedge (\mathsf{V}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, \pi') = 1) \wedge (pk_{\mathsf{Sign}} \notin \mathcal{PK}) \wedge (\mu = f_s(pk_{\mathsf{Sign}}))$;
  where Q is the set of statment-proof pairs and $\mathcal{PK}$ is the set of signature verification keys, both generated by $\mathsf{O}(x)$.

Lemma 4. *If the underlying one-time signature scheme is strongly unforgeable, and Groth's scheme guarantees knowledge-soundness, then* $\Pr[\mathsf{EXP}_2] \leq \Pr[\mathsf{EXP}_1] + \mathsf{negl}(\lambda)$.

Proof. We note that if $(x, \pi') \notin Q$ and "pk_{Sign} from (x, π'), has been generated by $\mathsf{O}(\cdot)$", then the (x, μ, π) is a valid message/signature pairs. Therefore by unforgeability of the signature scheme, we know that $(x, \pi) \notin Q$ and "pk_{Sign} has been generated by $\mathsf{O}(\cdot)$" happens with negligible probability, which allows us to focus on $pk_{\mathsf{Sign}} \notin \mathcal{PK}$.

Now, due to knowledge-soundness of the original scheme (there is an extractor $\mathcal{E}_{\mathcal{A}}$ where can extract witness from $\mathcal{A}$), if some witness is valid for $\mathcal{L}'$ and $(x, w) \notin \mathcal{R}_{\mathcal{L}}$, so we conclude it must be the case that there exists some s', such that ρ is valid commitment of s' and $\mu = f_{s'}(pk_{\mathsf{Sign}})$, which by perfectly binding property of the commitment scheme, it implies $\mu = f_s(pk_{\mathsf{Sign}})$. ∎

$\underline{\mathsf{EXP}_3}$(simulator):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s, r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s, r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s, r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x)$: $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (\mathsf{ts} \,\|\, (s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $(x, \pi') \leftarrow \mathcal{A}^{\mathsf{O}(x)}(\mathsf{crs}')$.
- Parse $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$; $w \leftarrow \mathcal{E}_{\mathcal{A}}(\mathsf{crs}', x, \pi, \mathsf{z}_{\mathcal{R}})$.
- Return 1 iff $((x, \pi') \notin Q) \wedge (\mathsf{V}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, \pi') = 1) \wedge (pk_{\mathsf{Sign}} \notin \mathcal{PK}) \wedge (\mu = f_s(pk_{\mathsf{Sign}}))$;
  where Q is the set of statment-proof pairs and $\mathcal{PK}$ is the set of signature verification keys, both generated by $\mathsf{O}(x)$.

Lemma 5. *If Groth's zk-SNARK guarantees zero-knowledge, then for two experiments* EXP_3 *and* EXP_2*, we have* $\Pr[\mathsf{EXP}_3] \leq \Pr[\mathsf{EXP}_2] + \mathsf{negl}(\lambda)$.

Proof. As the original scheme ensures (perfect) zero-knowledge, so it implies no polynomial time adversary can distinguish a proof generated by the simulator from a proof that is generated by the prover. So, as we are running in polynomial time, thus two experiments are indistinguishable. ∎

$\underline{\mathsf{EXP}_4}$(separating secret key of pseudo random function):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; $s', s, r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s', r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (s, s', r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x)$: $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = f_s(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (\mathsf{ts} \,\|\, (s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $(x, \pi') \leftarrow \mathcal{A}^{\mathsf{O}(x)}(\mathsf{crs}')$.
- Parse $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$; $w \leftarrow \mathcal{E}_{\mathcal{A}}(\mathsf{crs}', x, \pi, \mathsf{z}_{\mathcal{R}})$.
- Return 1 iff $((x, \pi') \notin Q) \wedge (\mathsf{V}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, \pi') = 1) \wedge (pk_{\mathsf{Sign}} \notin \mathcal{PK}) \wedge (\mu = f_s(pk_{\mathsf{Sign}}))$;
  where Q is the set of statment-proof pairs and $\mathcal{PK}$ is the set of signature verification keys, both generated by $\mathsf{O}(x)$.

Lemma 6. *If the commitment scheme used in the CRS generation is computationally hiding, then for two experiments* EXP_4 *and* EXP_3*, we have* $\Pr[\mathsf{EXP}_4] \leq \Pr[\mathsf{EXP}_3] + \mathsf{negl}(\lambda)$.

Proof. Computationally hiding of a commitment scheme implies that $\mathsf{Com}(m_1, r)$ and $\mathsf{Com}(m_2, r)$ are computationally indistinguishable. So this concludes the lemma. ∎

$\underline{\mathsf{EXP}_5}$(replace pseudo random function $f_s(\cdot)$ with true random function $F(\cdot)$):

- *Setup:* Sample $(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}})$; s', ~~s~~, $r \leftarrow_u \{0,1\}^\lambda$; $\rho := \mathsf{Com}(s', r)$; and output $(\mathsf{crs}' \,\|\, \mathsf{ts}') := ((\mathsf{crs}, \rho) \,\|\, (\mathsf{ts}, (~~s~~, s', r)))$; where ts' is simulation trapdoor.
- *Define function* $\mathsf{O}(x)$: $(pk_{\mathsf{Sign}}, sk_{\mathsf{Sign}}) \leftarrow \mathsf{KGen}(1^\lambda)$; set $\mu = F(pk_{\mathsf{Sign}})$; generate $\pi \leftarrow \mathsf{S}(\mathcal{R}_{\mathcal{L}'}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}, (x, \mu, pk_{\mathsf{Sign}}, \rho), (\mathsf{ts} \,\|\, (s, r)))$; sign $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{Sign}}, (x, \mu, \pi))$; return $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$.
- $(x, \pi') \leftarrow \mathcal{A}^{\mathsf{O}(x)}(\mathsf{crs}')$.
- Parse $\pi' := (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$; $w \leftarrow \mathcal{E}_{\mathcal{A}}(\mathsf{crs}', x, \pi, \mathsf{z}_{\mathcal{R}})$.
- Return 1 iff $((x, \pi') \notin Q) \wedge (\mathsf{V}'(\mathcal{R}_{\mathcal{L}}, \mathsf{z}_{\mathcal{R}}, \mathsf{crs}', x, \pi') = 1) \wedge (pk_{\mathsf{Sign}} \notin \mathcal{PK}) \wedge (\mu = F(pk_{\mathsf{Sign}}))$;
  where Q is the set of statment-proof pairs and $\mathcal{PK}$ is the set of signature verification keys, both generated by $\mathsf{O}(x)$.

Table II

A comparison of Groth's [6] and Groth-Maller [7] with the proposed scheme for arithmetic circuit satisfiability with m_0 elements instance, m wires, n MUL gates. Since [7] uses squaring gates, so n MUL gates translate to $2n$ squaring gates. Implementations are done on a PC with 3.40 GHz Intel Core i7-4770 CPU, in single-threaded mode, for an R1CS instance with $n = 10^6$ constraints and $m = 10^6$ variables, of which $m_0 = 10$ are input variables. $\mathbb{G}_1$ and $\mathbb{G}_2$: group elements, E: exponentiations and P: pairings. Note that in result of defining new language, new statement contains $(x, \mu, pk_{\mathsf{Sign}}, \rho)$ which has 3 new elements $(\mu, pk_{\mathsf{Sign}}, \rho)$, so $m'_0 = m_0 + 3$. All asymptotic analysis of our construction are done based on instantiation of commitment and pseudo random function with SHA-256 hash function. With this in mind, since new changes will add $\approx 50 \times 10^3$ MUL gates to n and m, so $n' = n + 50.000$ and $m' = m + 50.000$. Based on these changes, by considering the asymptotic performance of each algorithm in Groth's scheme [6], we write down the expected empirical performance of new scheme. For instance since $\approx 50 \times 10^3$ is 5% of 10^6 and CRS size is $O(m)$, so the CRS size and CRS generation time for new construction would be about 215 MB (1.05×201 MB < 215 MB) and 77 seconds (1.05×72.5 seconds < 77 seconds), respectively.

zk-SNARK in the CRS model	CRS Size, CRS Time	Proof Size	Prover Computation	Verifier Comp.	Security
Groth and Maller [7]	$m + 4n + 5$ $\mathbb{G}_1$	2 $\mathbb{G}_1$	$m + 4n - m_0$ E_1	m_0 E_1	Simulation Extractability
&	$2n + 3$ $\mathbb{G}_2$	1 $\mathbb{G}_2$	$2n$ E_2	5 P	
Empirical performance in libsnark	385 MB, 100.4 seconds	127 bytes	116.4 seconds	2.3 millisec.	
Groth [6]	$m + 2n - m_0$ $\mathbb{G}_1$	2 $\mathbb{G}_1$	$m + 3n - m_0 + 3$ E_1	m_0 E_1	Knowledge Soundness
&	$n + 3$ $\mathbb{G}_2$	1 $\mathbb{G}_2$	$n + 1$ E_2	3 P	
Empirical performance in libsnark	201 MB, 72.5 seconds	127 bytes	84.0 seconds	1.3 millisec.	
New variation **(with our instantiation)**	$m' + 2n' - m'_0 + 5$ $\mathbb{G}_1$	3 $\mathbb{G}_1$ + 2 $\mathbb{G}_2$	$m' + 3n' - m'_0 + 4$ E_1	$m'_0 + 1$ E_1	Simulation Extractability
&	$n' + 3$ $\mathbb{G}_2$	1 λ-bit string	$n' + 2$ E_2	4 P	
Empirical performance prediction	215 MB, $\approx$ 77 seconds	$<$ 200 bytes	$\approx$ 90 seconds	$\approx$ 1.8 millisec.	

Lemma 7. *If the underlying truly random function $F(\cdot)$ is secure, then* $\Pr[\mathsf{EXP}_4] \leq \Pr[\mathsf{EXP}_5]$.

Proof. By assuming function $F(\cdot)$ is secure, we can conclude no polynomial time adversary can distinguish an output of the true random function $F(\cdot)$ from an output of the pseudo random function $f_s(\cdot)$. Indeed, experiment EXP_5 can be converted to an adversary for the game of a *true random function*. ■

Claim 1. *For experiment* EXP_5*, we have* $\Pr[\mathsf{EXP}_5] \leq 2^{-\lambda}$.

Proof. From verification we know $pk_{\mathsf{Sign}} \notin \mathcal{PK}$, therefore $F(pk_{\mathsf{Sign}})$ has not been queried already. Thus, we will see $F(pk_{\mathsf{Sign}})$ as a newly generated random string independent from μ, which implies adversary only can guess. ■

This completes proof of the main theorem. ■

IV. Instantiation and Efficiency Evaluation

As we observed in Sec. III-A, defining the new language $\mathcal{L}'$ implies some changes in the algorithms of the original scheme. In this section, we discuss how efficient can be such changes (shown in Fig. 1). We first discuss how the used primitives can be efficiently instantiated and then go trough the efficiency of whole protocol.

Recall that in result of new changes, one need a pseudo random function, a commitment scheme and a one-time secure signature scheme. In similar practical cases, both pseudo random function and commitment scheme are instantiated using an efficient SHA-256 circuit which costs about $\approx 25 \times 10^3$ gates for one block (512-bit input) [10], [11][4]. We instantiate the signature scheme with Boneh and Boyen's signature [22] where its setup phase, signing and verification for message m can be summarized as below.

Setup: Given system parameters $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e}, [1]_1, [1]_2)$, randomly selects $sk \leftarrow_u \mathbb{Z}_p^*$, and computes $sk \cdot [1]_1$ and returns $(pk, sk) := ([sk]_1, sk)$.

Sining: Given a secret key sk, and a message m, computes $[\sigma]_2 = \left[\frac{1}{m + sk}\right]_2$ and returns $[\sigma]_2$ as the signature.

Verification: Given a public key pk, a message m, and a signature σ, verifies if $[m + sk]_1 \bullet \left[\frac{1}{m + sk}\right]_2 = [1]_T$.

where $\bullet$ denotes the paring operation. In our case, we use the same bilinear group as in zk-SNARK and m would be the hash (e.g with SHA-256) of concatenation of the proof elements with the statement, e.g. $m := H(x \| z_0 \| \pi)$. As it can be seen, the scheme generates a single-element signature in $\mathbb{G}_2$, its public key is a single group element in $\mathbb{G}_1$, and above all its verification only requires one paring. Note that $[1]_T$ can be prprocessed and shared in CRS.

So by considering the above instantiation new proof $\pi' = (\mu, \pi, pk_{\mathsf{Sign}}, \sigma)$ will be as $\pi' = (\mu, \pi, [sk]_1, \left[\frac{1}{m + sk}\right]_2)$ where from original scheme $\pi = ([a]_1, [b]_2, [c]_1)$, and μ is an output of the pseudo random function $f_s(\cdot)$ which usually is instantiated with SHA-256 hash function [11]. As a result, new proof will be 3 elements from $\mathbb{G}_1$, 2 elements from $\mathbb{G}_2$ and one 256-bit string. Consequently, verification of the proposed construction will add only one paring to the verification of original scheme that requires 3 parings. To the best of our knowledge, this is the first simulation-extractable zk-SNARKs in the CRS model which its verification requires 4 parings.

Next we evaluate efficiency of the new construction (given in Fig. 1) from different points of view. Tab. II summarize asymptotic and empirical performance of Groth's [6], Groth-Maller [7] zk-SNARKs[5] and evaluation of new construction proposed in Sec. III-A. The comparison is done for arithmetic circuit satisfiability with $m_0 = 10$ elements instance, $m = 10^6$ wires and $n = 10^6$ MUL gates [5]. All evaluations for the proposed construction are done for an arithmetic circuit satisfiability with the same parameters and empirical analysis are predicted based on asymptotic and implementation of

[4]More precisely, it has 25.538 gates in `xjsnark` library; available on https://github.com/akosba/xjsnark

[5]Based on analysis in `libsnark` library, https://github.com/scipr-lab/libsnark/tree/master/libsnark/zk_proof_systems/ppzksnark

Groth's zk-SNARK [6]

As an instance, since for a circuit with size n, the prover does $O(n)$ cryptographic and $O(n \log n)$ non-cryptographic computations, so one can observe that for circuits with 1×10^6 gates, by considering 50×10^3 new MUL gates, the overload would be around 5%, which by considering prover's execution time of Groth scheme (84.1 seconds), proof generation (Pro. Time) for new construction would take around 90 seconds. Similarly, for the considered instance in Tab. II, the CRS Size in the new scheme will be around 215 Mega Bytes (MB) which is considerably smaller than the public parameters of Groth-Maller zk-SNARK [7]. As it can be concluded from the evaluations in Tab. II, in order to give non-malleable proofs for an arithmetic circuit satisfiability in circuits with large number of MUL gates (clearly larger than 50×10^3) the proposed variation of Groth's zk-SNARK can outperform Groth-Maller zk-SNARK. Note that, however new construction has slightly larger proof size than Groth-Maller zk-SNARK, but still its verification would be faster than Groth-Maller zk-SNARK.

V. Conclusion

In this paper, we proposed a variation of the state-of-the-art zk-SNARK proposed by Groth [6], that can achieve simulation extractability. Simulation extractability is an amplified version of knowledge-soundness that additionally guarantees non-malleability of proofs. We used the known OR technique to define a new language $\mathcal{L}'$ from the language $\mathcal{L}$ in original scheme, that led to apply some changes in algorithms of the original scheme. Evaluations showed that in practical systems, e.g. when a prover needs to give a proof for satisfiability of an arithmetic circuit with 10^6 MUL gates, 10^6 variables of which 10 are input variables, new changes have minimal effect on the efficiency of the original zk-SNARK which considerably is optimized. Strictly speaking, our comparisons showed that for arithmetic circuits with approximately larger than 50×10^3 MUL gates, the proposed construction can outperform Groth-Maller SE zk-SNARK [7] that currently is the only zk-SNARK in the CRS model which achieves simulation extractability. We emphasize that in current real-life systems that use zk-SNARK, the underlying arithmetic circuits have much more number of gates than 50×10^3. For instance, in Zerocash cryptocurrency [10] their current circuit for *pour* transactions has 2×10^6 MUL gates; or similarly in Hawk smart contract system [11], their circuit for *finalize* operation in an auction with 50 bidders has around 4×10^6 MUL gates.

In comparison with Groth-Maller SE zk-SNARK, we observed that in our case the proof size is extended slightly, but still its total size is less than 200 bytes for 128-bit security; and interestingly its verification is faster than proof verification in their scheme.

References

[1] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications (extended abstract)," in *20th ACM STOC*. ACM Press, May 1988, pp. 103–112.

[2] J. Groth, "Short pairing-based non-interactive zero-knowledge arguments," in *ASIACRYPT 2010*, ser. LNCS, M. Abe, Ed., vol. 6477. Springer, Heidelberg, Dec. 2010, pp. 321–340.

[3] H. Lipmaa, "Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments," in *TCC 2012*, ser. LNCS, R. Cramer, Ed., vol. 7194. Springer, Heidelberg, Mar. 2012, pp. 169–189.

[4] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *2013 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2013, pp. 238–252.

[5] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive arguments for a von neumann architecture," Cryptology ePrint Archive, Report 2013/879, 2013, http://eprint.iacr.org/2013/879.

[6] J. Groth, "On the size of pairing-based non-interactive arguments," in *EUROCRYPT 2016, Part II*, ser. LNCS, M. Fischlin and J.-S. Coron, Eds., vol. 9666. Springer, Heidelberg, May 2016, pp. 305–326.

[7] J. Groth and M. Maller, "Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs," in *CRYPTO 2017, Part II*, ser. LNCS, J. Katz and H. Shacham, Eds., vol. 10402. Springer, Heidelberg, Aug. 2017, pp. 581–612.

[8] I. Damgård, "Towards practical public key systems secure against chosen ciphertext attacks," in *CRYPTO'91*, ser. LNCS, J. Feigenbaum, Ed., vol. 576. Springer, Heidelberg, Aug. 1992, pp. 445–456.

[9] C. Gentry and D. Wichs, "Separating succinct non-interactive arguments from all falsifiable assumptions," in *43rd ACM STOC*, L. Fortnow and S. P. Vadhan, Eds. ACM Press, Jun. 2011, pp. 99–108.

[10] 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*. IEEE Computer Society Press, May 2014, pp. 459–474.

[11] 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*. IEEE Computer Society Press, May 2016, pp. 839–858.

[12] A. Juels, A. E. Kosba, and E. Shi, "The ring of Gyges: Investigating the future of criminal smart contracts," in *ACM CCS 16*, E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, Eds. ACM Press, Oct. 2016, pp. 283–295.

[13] K. Baghery, "On the efficiency of privacy-preserving smart contract systems," Cryptology ePrint Archive, Report 2019/480, 2019, https://eprint.iacr.org/2019/480.

[14] J. Groth, "Simulation-sound NIZK proofs for a practical language and constant size group signatures," in *ASIACRYPT 2006*, ser. LNCS, X. Lai and K. Chen, Eds., vol. 4284. Springer, Heidelberg, Dec. 2006, pp. 444–459.

[15] B. Abdolmaleki, K. Baghery, H. Lipmaa, and M. Zajac, "A subversion-resistant SNARK," in *ASIACRYPT 2017, Part III*, ser. LNCS, T. Takagi and T. Peyrin, Eds., vol. 10626. Springer, Heidelberg, Dec. 2017, pp. 3–33.

[16] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai, "Robust non-interactive zero knowledge," in *CRYPTO 2001*, ser. LNCS, J. Kilian, Ed., vol. 2139. Springer, Heidelberg, Aug. 2001, pp. 566–598.

[17] A. E. Kosba, Z. Zhao, A. Miller, Y. Qian, T. H. Chan, C. Papamanthou, R. Pass, A. Shelat, and E. Shi, "C∅C∅: A Framework for Building Composable Zero-Knowledge Proofs," Tech. Rep. 2015/1093, Nov. 10, 2015, http://eprint.iacr.org/2015/1093, last accessed version from 9 Apr 2017.

[18] S. Bowe and A. Gabizon, "Making groth's zk-snark simulation extractable in the random oracle model," *IACR Cryptology ePrint Archive*, vol. 2018, p. 187, 2018. [Online]. Available: http://eprint.iacr.org/2018/187

[19] J. Kim, J. Lee, and H. Oh, "Qap-based simulation-extractable snark with a single verification," Cryptology ePrint Archive, Report 2019/586, 2019, https://eprint.iacr.org/2019/586.

[20] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, "Quadratic span programs and succinct NIZKs without PCPs," in *EUROCRYPT 2013*, ser. LNCS, T. Johansson and P. Q. Nguyen, Eds., vol. 7881. Springer, Heidelberg, May 2013, pp. 626–645.

[21] N. Bitansky, R. Canetti, O. Paneth, and A. Rosen, "On the existence of extractable one-way functions," in *46th ACM STOC*, D. B. Shmoys, Ed. ACM Press, May / Jun. 2014, pp. 505–514.

[22] D. Boneh and X. Boyen, "Short signatures without random oracles and the SDH assumption in bilinear groups," *Journal of Cryptology*, vol. 21, no. 2, pp. 149–177, Apr. 2008.