Avoiding Random Oracles in Bowe–Gabizon SNARK Verification

Vanishree Rao and Izaak Meckler

O(1) Labs

May 2019

1 Introduction

This note describes a modification of the Bowe–Gabizon simulation-extractable SNARK [2] (hereafter, we will refer to the Bowe–Gabizon SNARK as the BG18 SNARK with the following additional property: Given a BG18 proof π, we can easily compute a merely *knowledge-sound* SNARK which is more efficient to verify in certain contexts.

Specifically, consider the following execution environments, which have differing security requirements and efficiency characteristics:

- **Environment 1:** Simulation-extractability of proofs is required and invoking a random oracle is inexpensive.
- **Environment 2:** Simulation-extractability of proofs is not required (i.e., knowledge-soundness is sufficient) and invoking a random oracle is expensive.

The BG18 construction gives a SNARK whose proofs can be efficiently verified in both environments. That is, if one is willing to use a random oracle, one can get a simulation-extractability guarantee about a given proof, while at the same time a proof can be checked "up to knowledge-soundness" without having to invoke a random oracle.

To further motivate the construction, let us describe the two environments as they occur in the Coda protocol [1]. The first environment corresponds to the normal execution of a CPU (where random oracle instantiations like blake2 are efficient) verifying another party's proofs from the network.

The second environment corresponds to verifying proofs within a rank-1 constraint system while performing proof composition. In our context, simulation-extractability is not important here while random oracle instantiations like blake2 are very inefficient to implement.

For an intuitive summary of the construction, note that the BG18 SNARK is a modification of the Groth's simulation-sound SNARK [3] (hereafter, we will refer to the Groth's SNARK as the Groth16 SNARK) in two ways:

1. In the Groth16 SNARK, the common random string (CRS) contains a g^{δ} term which is used at multiple points of computing a snark. The modification is that wherever g^{δ} is used, it is replaced with $g^{\delta'}$ for $\delta' = d\delta$ for some scalar d known to the prover.

2. Provide an additional argument of knowledge of d.

Our modification pertains specifically to the second item above. Specifically, we present a simplified argument of knowledge of the discrete log. While the BG18 version used a fresh random group element in constructing the DL argument of knowledge (namely, the output of a random oracle on the part of the proof resulting from the first item above), we will use a uniformly random but fixed group element.

2 Notations and Definitions

Notations.

- We denote that an element s is sampled uniformly at random from a set S by $s \leftarrow S$.
- We denote that a probabilistically polynomial-time algorithm $\mathcal{A}$ runs on an input x with random coins r and outputs y by $y \leftarrow \mathcal{A}(x; r)$. Often, we omit specifying r.
- Like in [3], we write $(y; z) \leftarrow (\mathcal{A}||\mathcal{E})(x)$ when $\mathcal{A}$ on input x outputs y, and $\mathcal{E}_{\mathcal{A}}$ on the same input (including random coins) outputs z.

Quadratic Arithmetic Programs. Like in [3], we use QAPs as a model of computation. Specifically, we will consider a degree n and size m QAP, $\mathcal{Q} = (\{u_i(X), v_i(X), w_i(X)\}_{i=0}^{m}, t(X))$ with a corresponding relation $R = \{((a_1, \ldots, a_\ell), (a_{\ell+1}, \ldots, a_m))\} \subseteq \mathbb{F}^\ell \times \mathbb{F}^{m-\ell}$ such that, with $a_0 = 1$,

$$\sum_{i=0}^{m} a_i u_i(X) \cdot \sum_{i=0}^{m} a_i v_i(X) = \sum_{i=0}^{m} a_i w_i(X) + h(X)t(X)$$

for some $n-2$ degree quotient polynomial $h(X)$, where n is the degree of $t(X)$.

NIZK Argument of Knowledge. A NIZK argument of knowledge consists of the following set of algorithms.

$(\sigma, \tau) \leftarrow \mathsf{Setup}(R)$: Setup, on input a relation R, outputs a common reference string σ and a trapdoor τ.

$\pi \leftarrow \mathsf{Prove}(R, \sigma, \phi, \omega)$: Prove takes as input $(\phi, \omega) \in R$, besides R, σ, and outputs a proof π.

$\mathsf{acc/rej} \leftarrow \mathsf{Vf}(R, \sigma, \phi, \pi)$: Vf takes as input R, σ, ϕ, π and outputs the accept symbol acc or the reject symbol rej.

$\pi \leftarrow \mathsf{Sim}(R, \sigma, \tau, \phi)$: Sim takes as input R, σ, τ, ϕ and outputs a proof π.

Definition 1 (NIZK Argument of Knowledge) *A system* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vf}, \mathsf{Sim})$ *is a perfect non-interactive zero-knowledge argument of knowledge for relation R if it has perfect completeness, perfect zero-knowledge and computational knowledge soundness as defined below. Let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda \in \mathbb{N}}$ be the set of all polynomial-time decidable binary relations*

Perfect completeness. *Completeness says that, for any given true statement, an honest prover knowing its witness should be able to convince an honest verifier. For all $\lambda \in \mathbb{N}$, $R \in \mathcal{R}_\lambda$, $(\phi, \omega) \in R$,*

$$\Pr[(\sigma, \tau) \leftarrow \mathsf{Setup}(R); \pi \leftarrow \mathsf{Prove}(R, \sigma, \phi, \omega) : \mathsf{Vf}(R, \sigma, \phi, \pi) = \mathsf{acc}] = 1$$

Perfect zero-knowledge. *An argument is zero-knowledge if it does not leak any information besides the truth of the statement. We say* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vf}, \mathsf{Sim})$ *is perfect zero-knowledge if for all $\lambda \in \mathbb{N}$, $R \in \mathcal{R}_\lambda$, $(\phi, \omega) \in R$, and all adversaries $\mathcal{A}$,*

$$\begin{aligned} &\Pr[(\sigma, \tau) \leftarrow \mathsf{Setup}(R); \pi \leftarrow \mathsf{Prove}(R, \sigma, \phi, \omega) : \mathcal{A}(R, \sigma, \tau, \pi) = 1] \\ =&\Pr[(\sigma, \tau) \leftarrow \mathsf{Setup}(R); \pi \leftarrow \mathsf{Sim}(R, \tau, \phi) : \mathcal{A}(R, \sigma, \tau, \pi) = 1] \end{aligned}$$

Computational knowledge soundness. *The system is said to be knowledge sound or an argument of knowledge if there is an extractor that can compute a witness whenever the adversary produces a valid argument. The extractor gets full access to the adversary's state, including any random coins. Formally, it is required that for all non-uniform polynomial time adversaries $\mathcal{A}$ there exists a non-uniform polynomial time extractor $\mathcal{E}$ such that,*

$$\begin{aligned} \Pr[R \leftarrow \mathcal{R}_\lambda; (\sigma, \tau) \leftarrow \mathsf{Setup}(R); ((\phi, \pi); \omega) \leftarrow (\mathcal{A} || \mathcal{E}_\mathcal{A})(R, \sigma) \\ (\phi, \omega) \notin R \text{ and } \mathsf{Vf}(R, \sigma, \phi, \pi) = \mathsf{acc}] \approx 0 \end{aligned}$$

3 Constant-base-DL-proof BG18

In this section we describe a variant of the BG18 SNARK. Noting that BG18 had Groth16 as its starting point, we too shall present our construction as a

variant of Groth16 for readability. The proposed SNARK is specified by the following algorithms.

We will denote the Groth16 SNARK system by $(\mathsf{Setup}^{(1)}, \mathsf{Prove}^{(1)}, \mathsf{Vf}^{(1)}, \mathsf{Sim}^{(1)})$. We will denote a DL non-interactive argument of knowledge by $(\mathsf{Setup}^{(2)}, \mathsf{Prove}^{(2)}, \mathsf{Vf}^{(2)})$. We highlight the parts differing from the Groth16 SNARK in blue.

$(\sigma, \tau) \leftarrow \mathsf{Setup}(R)$:

Run $(\sigma^{(1)}, \tau^{(1)}) \leftarrow \mathsf{Setup}^{(1)}(R)$. Also, run $(\sigma^{(2)}, \tau^{(2)}) \leftarrow \mathsf{Setup}^{(2)}(R)$. Set $\sigma = (\sigma^{(1)}, \sigma^{(2)})$ and $\tau = (\tau^{(1)}, \tau^{(2)})$. Output (σ, τ).

Recall that $\sigma^{(1)}$ is of the following form: $\sigma^{(1)} = ([\sigma_1]_1, [\sigma_2]_2)$, where

$$\sigma_1 = \left(\begin{array}{c} \alpha, \beta, \delta, \{x^i\}_{i\in[0\ldots 2n-2]}, \{\alpha x^i\}_{i\in[1\ldots n-1]}, \{\beta x^i\}_{i\in[1\ldots n-1]} \\ \left\{\frac{\beta u_i(x)+\alpha v_i(x)+w_i(x)}{\delta}\right\}_{i\in[\ell+1\ldots m]}, \left\{\frac{x^i t(x)}{\delta}\right\}_{i\in[0\ldots n-2]} \end{array} \right)$$

and

$$\sigma_2 = \left(\beta, \delta, \{x^i\}_{i\in[0\ldots n_1]}\right)$$

$\pi \leftarrow \mathsf{Prove}(R, \sigma, (a_1, \ldots, a_\ell), (a_{\ell+1}, \ldots, a_m))$:

1. Choose $d \leftarrow \mathbb{F}_p^*$ uniformly at random. Define $\delta' = d\delta$.
2. Choose $r, s \leftarrow \mathbb{F}_p^*$ uniformly at random and compute $\pi^{(1)} = ([A]_1, [B]_2, [C]_1, [\delta']_2)$, where

$$\begin{aligned} A &= \alpha + \sum_{i=0}^{m} a_i u_i(x) + r\delta' \\ B &= \beta + \sum_{i=0}^{m} a_i v_i(x) + s\delta' \\ C &= \frac{\sum_{i=\ell+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + h(x)t(x)}{\delta'} + As + Br - rs\delta' \end{aligned}$$

3. Compute $\pi^{(2)} = \mathsf{Prove}^{(2)}(\sigma^{(2)}, ([\delta]_2, [\delta']_2), d)$.

Output $\pi = (\pi^{(1)}, \pi^{(2)})$.

$\mathsf{acc/rej} \leftarrow \mathsf{Vf}(R, \sigma, (a_1, \ldots, a_\ell), \pi)$:

Parse $\pi = (\pi^{(1)}, \pi^{(2)})$ and $\pi^{(1)} = ([A]_1, [B]_2, [C]_1, [\delta']_2)$ and check that

$$[A]_1 \cdot [B]_2 = [\alpha]_1 \cdot [\beta]_2 + \sum_{i=0}^{\ell} a_i \left[\frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\gamma}\right]_1 \cdot [\gamma]_2 + [C]_1 \cdot [\delta']_2 \tag{1}$$

and

$$\mathsf{Vf}_2(\sigma^{(2)}, ([\delta]_2, [\delta']_2), \pi^{(2)}) = \mathsf{acc} \tag{2}$$

$\pi \leftarrow \mathsf{Sim}(R, \sigma, \tau, (a_1, \ldots, a_\ell))$:

1. Choose $d \leftarrow \mathbb{F}_p^*$ and compute $\delta' = d\delta$.
2. Create σ_1' by replacing δ in σ_1 with δ'.
3. $([A]_1, [B]_2, [C]_1) \leftarrow \mathsf{Sim}_1(\sigma_1')$.
4. compute $[\delta']_2$ and set $\pi^{(1)} = ([A]_1, [B]_2, [C]_1, [\delta']_2)$.
5. Compute $\pi^{(2)} = \mathsf{Prove}^{(2)}(\sigma^{(2)}, ([\delta]_2, [\delta']_2), d)$.
6. Output $\pi = (\pi^{(1)}, \pi^{(2)})$.

4 Proof of Security

Theorem 1 *The variable-δ Groth16 system is a non-interactive zero-knowledge argument with perfect completeness and perfect zero-knowledge. Assuming that the knowledge of exponent assumption holds, it has statistical knowledge soundness against adversaries that only use a polynomial number of generic bilinear group operations.*

Proof: It is easy to see that the system satisfies perfect completeness.

Perfect zero-knowledge. Note that the second part of the proof is not zero knowledge in d. However, intuitively, the additional random elements r, s in A, B render them distributed uniformly random subjected to satisfying the first verification equation with C. In fact, Sim first samples d and then computes the other elements of the proof accordingly. It is easy to see that the distribution of Sim's output is identical to that of Prove.

Statistical knowledge soundness. As shown in [3], it suffices to show that for the corresponding proof system with only the exponents, the system is statistically knowledge sound against affine adversarial prover strategies. We will now show that for any such adversarial prover $\mathcal{A}$, we can extract the witness with non-negligible probability.

Firstly, we will argue extractability of d. Then, we will argue that the elements in $\sigma^{(1)}$ that are not in [3] do not unduly "help" the adversarial prover in computing a proof. Finally, we briefly argue how the proof of security in [3] completes the rest of our proof.

Extractability of d. Since $(\mathsf{Setup}^{(2)}, \mathsf{Prove}^{(2)}, \mathsf{Vf}^{(2)})$ is an argument of knowledge, for every adversarial prover, there exists an extractor $\mathcal{E}^{(2)}$ that can extract from the prover the discrete log d of $[\delta']_2$ with respect to $[\delta]_2$. Hence, going forward, we can replace all the arguments where δ' is involved with $d\delta$ where d is known. This implies that d can simply be baked into the scalars of the equations

of affine relations considered in [3], such as Equation (3) below; hence, similar arguments hold true for the proposed proof system too.

Prover cannot use additional CRS terms. Since $\mathcal{A}$ performs only affine operations on the elements in the CRS, we have

$$\begin{aligned} A =& A_\alpha \alpha + A_\beta \beta + A_\delta \delta + A_\alpha(x)\alpha + A_\beta(x)\beta + A(x) \\ &+ \sum_{i=\ell+1}^{m} A_i \frac{(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta} + A_t(x)\frac{t(x)}{\delta} \end{aligned} \tag{3}$$

where, $A_\alpha, A_\beta, A_\delta, A_i$ are known field elements, $A_\alpha(X), A_\beta(X)$ are known $n-1$ degree polynomials without the constant term, $A(X)$ is a known $2n-2$ degree polynomial and $A_t(X)$ is a known $n-2$ degree polynomial.

Parse $A(X) = A_{\mathsf{low}}(X) + A_{\mathsf{high}}(X)$, where $A_{\mathsf{low}}(X)$ contains all the terms with degree $i \leq n-1$.

A majority of the proof is similar to [3], except that we need to account for new terms in the CRS, namely, $\{x^i\}_{i\in[n\ldots 2n-2]}, \{\alpha x^i\}_{i\in[1\ldots n-1]}$ and $\{\beta x^i\}_{i\in[1\ldots n-1]}$, that were not in the CRS in [3].

We will use the following terminology like in [3]. We will consider monomials that are quotients of two monomials; e.g., $\frac{x}{\delta}$. By $M \in A$, we denote that the monomial M appears with non-zero coefficient when A is written as a unique non-linear combination of the monomials in α, β, δ, x.

Since $\alpha\beta \notin \sigma^{(1)}$ and $\alpha\beta \notin \sigma^{(2)}$ and since $\alpha\beta \in AB$, we have that either $\alpha \in A, \beta \in B$ or $\alpha \in B, \beta \in A$; let $\alpha \in A, \beta \in B$ without loss of generality. Hence, $A_\alpha = 1, B_\beta = 1, A_\beta = 0, B_\alpha = 0$.

Now we will show by contradiction that $A_\alpha(X), A_\beta(X), B_\alpha(X), B_\beta(X)$ are zero polynomials. If $\alpha A_\alpha(X) \in A$, then since $\beta \in B$, $\alpha\beta A_\alpha(X) \in AB$. Denoting the three terms on the right hand side of the first verification equation (namely, Equation (1)) as T_1, T_2, T_3 respectively, we have that clearly $\alpha\beta A_\alpha(X) \notin T_1$ since $A_\alpha(X)$ has no constant term and clearly $\alpha\beta A_\alpha(X) \notin T_2$. Thus, $\alpha\beta A_\alpha(X) \in T_3$ which implies that $\frac{\alpha\beta A_\alpha(X)}{\delta} \in C$, which is a contradiction, since $\frac{\alpha\beta x^i}{\delta}$ is not in the CRS and hence the adversary could not have computed such a C. Hence, $A_\alpha(X)$ is a zero polynomial. Similar arguments show that $A_\beta(X), B_\alpha(X), B_\beta(X)$ are also zero polynomials.

Next, we will show by contradiction that the adversary cannot use $\{x^i\}_{i\in[n\ldots 2n-2]}$ with non-zero coefficients while constructing its proof. Assume for contradiction that $x^i \in A$ for some $i > n-1$. Since $\beta \in B$, we have that $\beta x^i \in AB$, which implies that it has to be in one of T_1, T_2, T_3. However, clearly $\beta x^i \notin T_1$ and since only powers of x lower than n make up T_2, we have that $\beta x^i \notin T_2$. Also, if $\beta x^i \in T_3$ then $\frac{\beta x^i}{\delta} \in C$, which is a contradiction since $\frac{\beta x^i}{\delta}$ occurs in the CRS only for $i \leq n-1$.

Witness extractability. Having proven that the additional items in the CRS do not unduly facilitate a prover and that d is extractable, witness extractability of the proposed system follows from that of Groth16 by simply absorbing d into the scalars of the affine relations considered in the security proof of Groth16.

□

5 An Instantiation of DL Non-interactive Argument of Knowledge

A non-interactive argument of knowledge is an argument system similar to a non-interactive zero-knowledge argument of knowledge defined in Definition 1, except that it does not need to satisfy the zero knowledge property. Specifically, a non-interactive argument of knowledge needs to satisfy perfect completeness and computational knowledge soundness.

We focus on proving knowledge of discrete log of an element in G_2 of a bilinear group, which will be useful in the proposed Variable-δ Groth16 SNARK. We propose to use the following simple discrete log non-interactive argument of knowledge, which simply involves exponentiating the same discrete log on a different random group generator. Specifically, the DL AoK $(\mathsf{Setup}^{(2)}, \mathsf{Prove}^{(2)}, \mathsf{Vf}^{(2)})$ is described as follows.

$(f_1, f_2) \leftarrow \mathsf{Setup}^{(2)}(p, G_1, G_2, G_T, e, g_1, h_2)$**:** On input a bilinear group description $(p, G_1, G_2, G_T, e, g_1, h_2)$, where g_1, h_2 are random generators of G_1, G_2, respectively, sample $r \leftarrow \mathbb{Z}_p^*$ and compute $f_1 = g_1^r$ and $f_2 = h_2^r$. Output $\sigma^{(2)} = (p, G_1, G_2, G_T, e, g_1, h_2, f_1, f_2)$.

$\pi^{(2)} \leftarrow \mathsf{Prove}(\sigma^{(2)}, x, d)$**:** Upon input the CRS, a statement of the form $x = h_2^d$ and the discrete log $d \in \mathbb{Z}_p^*$ of h_2^d with respect to h_2, compute f_2^d and output $\pi^{(2)} = f_2^d$.

$\mathsf{Vf}(\sigma^{(2)}, x, \pi^{(2)})$**:** Upon input the CRS, a statement and a proof, output acc if $e(f_1, h_2^d) = e(g_1, f_2^d)$ and rej otherwise.

Remark 1 *When instantiating the proposed SNARK with the above DL argument of knowledge, the Setup algorithm takes as input the same bilinear group description as used in the first part of the proposed SNARK, except that, instead of g_2, it takes $h_2 = g_2^\delta$ as the generator for G_2, since knowledge discrete log needs to be proven with respect to h_2 in the proposed SNARK.*

The proof of security follows in a straight-forward manner from the following knowledge of exponent assumption.

Assumption 1 *The knowledge of exponent assumption holds for $\mathcal{G}$ if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ there exists a non-*

uniform probabilistic polynomial-time extractor $\mathcal{E}$ so that

$$\begin{gathered}\Pr[(p, G_1, G_2, G_T, e) \leftarrow \mathcal{G}(1^\lambda); \\ h_1 \leftarrow G_1 \setminus \{1_1\}; h_2 \leftarrow G_2 \setminus \{1_2\}; \\ k \leftarrow \mathbb{Z}_p^*; K_1 \leftarrow h_1^k; K_2 \leftarrow h_2^k; \\ \textit{Set } \rho := (p, G_1, G_2, G_T, h_1, h_2, K_1, K_2); \\ (C_2, D_2; d) \leftarrow (\mathcal{A}||\mathcal{E})(\rho) : D_2 = C_2^k \textit{ and } C_2 \neq K_1^d] \approx 0\end{gathered}$$

Theorem 2 *Suppose that the knowledge of exponent assumption holds in a bilinear group. Then,* $(\mathsf{Setup}^{(2)}, \mathsf{Prove}^{(2)}, \mathsf{Vf}^{(2)})$ *is a DL non-interactive argument of knowledge for discrete logarithm of elements in* G_2*.*

Proof: Perfect completeness is straight-forward to observe. Extractability also directly follows from the knowledge of exponent assumption. □

6 Acknowledgements

* We thank Amit Sahai for helpful discussions.

References

[1] MS Windows NT kernel description. `https://codaprotocol.com`. Accessed: 2019-04-15.

[2] Sean Bowe and Ariel 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`.

[3] Jens Groth. On the size of pairing-based non-interactive arguments. *IACR Cryptology ePrint Archive*, 2016:260, 2016.