On the security of the BCTV Pinocchio zk-SNARK variant

Ariel Gabizon*

February 13, 2019

Abstract

The main result of this note is a severe flaw in the description of the zk-SNARK in [BCTV14]. The flaw stems from including redundant elements in the CRS, as compared to that of the original Pinocchio protocol [PHGR16], which are vital not to expose. The flaw enables creating a proof of knowledge for *any* public input given a valid proof for *some* public input. We also provide a proof of security for the [BCTV14] zk-SNARK in the generic group model, when these elements are excluded from the CRS, provided a certain linear algebraic condition is satisfied by the QAP polynomials.

1 Introduction

Parno et. al [PHGR16] presented a zk-SNARK construction based on the breakthrough work of [GGPR13] that they called Pinocchio. Ben-Sasson et. al [BCTV14] presented a variant of Pinocchio with the advantage of shorter verification time and verification key length, at the expense of an arguably negligible increase in prover running time. However, [BCTV14] did not present a security proof for this variant, and in fact Parno [Par15] found an attack against the [BCTV14] SNARK and suggested to mitigate it by imposing a certain linear independence condition on some of the public input polynomials – which [BCTV14] did in a revised version of the paper and corresponding implementation [lib]. In this note, we show a more severe attack on [BCTV14] that takes advantage of redundant elements in the proving key that should have been omitted.

1.1 Impacted work

The first proof of security for [BCTV14] was given in [BGG17]. However, the proof has an error and in fact we discovered the attack while reviewing it. Other papers that leverage the [BCTV14] construction inherit the flaw also; the ones we found are [BBFR15, Fuc18].

The (`libsnark` [lib]) implementation of the [BCTV14] prover distributed alongside the paper expects that the elements of concern are removed from the CRS, apparently intended as one of several optimizations that deviated from the construction as described in the paper. (The elements are not used by the prover.) As a result, parameters constructed directly by `libsnark` are not vulnerable to this specific attack.

However, the multi-party computation of [BGG17] follows the construction described in [BCTV14] closely and so the elements are produced during the protocol's execution, though they do not appear in the final parameters in order to be compatible with the `libsnark` prover. Unfortunately,

*This work was done while the author was working at the Zcash Company.

the transcript of the protocol's execution must be distributed so that the resulting parameters can be verified, and so the elements are revealed.

We also discovered an independent implementation of [BCTV14] which appears to inherit the flaw [sna], as unlike `libsnark` its key generation and proving routines match the paper closely.

Lastly, we emphasize that the mitigation of Parno [Par15] for the attack presented there, does not mitigate the attack presented here.

1.2 Notation

We will be working over bilinear groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_t$ each of prime order p, together with respective generators g_1, g_2 and g_T. These groups are equipped with a non-degenerate bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, with $e(g_1, g_2) = g_T$. We write $\mathbb{G}_1$ and $\mathbb{G}_2$ additively, and $\mathbb{G}_t$ multiplicatively. We denote by $\mathbb{F}$ the field of the same order p. For $a \in \mathbb{F}$, we denote $[a]_1 := a \cdot g_1, [a]_2 := a \cdot g_2$.

1.3 Description of [BCTV14] SNARK

We recall the zk-SNARK of [BCTV14] as described in that paper. We assume familiarity with quadratic arithmetic programs. See e.g., Section 2.3 in [Gro16] for definitions. We use similar notation to [BCTV14], denoting by m the size of the QAP, d the degree and n the number of public inputs. More specifically, our QAP has the form $\left\{\{A_i(X), B_i(X), C_i(X)\}_{i\in[0..m]}, Z(X)\right\}$ where $A_i, B_i, C_i \in \mathbb{F}[X]$ have degree at most[1] d, and $Z \in \mathbb{F}[X]$ has degree exactly d.

We proceed to describe the proving system of [BCTV14]. We assume we are already given a description of the groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$, the pairing e, and uniformly chosen generators $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$, and these are all public.

BCTV key generation:

1. Sample random $\tau, \rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \gamma, \beta \in \mathbb{F}^*$
2. For $i \in [0..d]$ output $\mathsf{pk}_{H,i} := \left[\tau^i\right]_1$
3. For $i \in [0..m]$ output
   (a) $\mathsf{pk}_{A,i} := [\rho_A A_i(\tau)]_1$
   (b) $\mathsf{pk}'_{A,i} := [\alpha_A \rho_A A_i(\tau)]_1$,
   (c) $\mathsf{pk}_{B,i} := [\rho_B B_i(\tau)]_2$,
   (d) $\mathsf{pk}'_{B,i} := [\alpha_B \rho_B B_i(\tau)]_1$,
   (e) $\mathsf{pk}_{C,i} := [\rho_A \rho_B C_i(\tau)]_1$,
   (f) $\mathsf{pk}'_{C,i} := [\alpha_C \rho_A \rho_B C_i(\tau)]_1$
   (g) $\mathsf{pk}_{K,i} := [\beta(\rho_A A_i(\tau) + \rho_B B_i(\tau) + \rho_A \rho_B C_i(\tau))]_1$
4. Output the additional verification key elements $([\alpha_A]_2, [\alpha_B]_1, [\alpha_C]_2, [\gamma]_2, [\beta\gamma]_1, [\beta\gamma]_2, [\rho_A \rho_B Z(\tau)]_2)$

[1] [BCTV14] define A_i, B_i, C_i to be of degree strictly less than d, however since Z needs to be later added to $\{A_i\}$,$\{B_i\}$,$\{C_i\}$ for zero-knowledge, it is more convenient for us to allow degree at most d and assume Z is already included. Note that in terms of the set of satisfiable instances $x \in \mathbb{F}^n$ every degree d QAP is equivalent to one where $\{A_i, B_i, C_i\}$ have degree smaller than d obtained by taking the original polynomials mod Z.

BCTV **prover:**

The prover P has in his hand a QAP solution $(x_0 = 1, x_1, \ldots, x_m)$ that coincides with the public input $x = (x_1, \ldots, x_n)$ and satisfies the following: If we define $A := \sum_{i=0}^{m} x_i \cdot A_i$, $B := \sum_{i=0}^{m} x_i \cdot B_i$, and $C := \sum_{i=0}^{m} x_i \cdot C_i$; then the polynomial $P := A \cdot B - C$ will be divisble by the target polynomial Z, and P can compute the polynomial H of degree at most d with $P = H \cdot Z$. Let $A_{\text{mid}} := A - \sum_{i=0}^{n} x_i \cdot A_i$.

Given the proving key, P computes as linear combinations of the proving key elements

1. $\pi_A := [\rho_A A_{\text{mid}}(\tau)]_1$, $\pi'_A := [\alpha_A \rho_A A_{\text{mid}}(\tau)]_1$.
2. $\pi_B := [\rho_B B(\tau)]_2$, $\pi'_B := [\alpha_B \rho_B B(\tau)]_1$.
3. $\pi_C := [\rho_A \rho_B C(\tau)]_1$, $\pi'_C := [\alpha_C \rho_A \rho_B C(\tau)]_1$.
4. $\pi_K := [\beta(\rho_A A(\tau) + \rho_B B(\tau) + \rho_A \rho_B C(\tau))]_1$.
5. $\pi_H := [H(\tau)]_1$.

and outputs $\pi = (\pi_A, \pi_B, \pi_C, \pi'_A, \pi'_B, \pi'_C, \pi_H, \pi_K)$,

BCTV **verifier:**

Denote the "public input component"

$$\mathsf{PI}(x) := \mathsf{pk}_{A,0} + \sum_{i=1}^{n} x_i \mathsf{pk}_{A,i} = \left[\rho_A A_0(\tau) + \sum_{i=1}^{n} x_i \rho_A A_i(\tau) \right]_1$$

The verifier, using pairings and the verification key, checks the following.

1. $e(\pi'_A, g_2) = e(\pi_A, [\alpha_A]_2)$.
2. $e(\pi'_B, g_2) = e([\alpha_B]_1, \pi_B)$.
3. $e(\pi'_C, g_2) = e(\pi_C, [\alpha_C]_2)$.
4. $e(\pi_K, [\gamma]_2) = e(\mathsf{PI}(x) + \pi_A + \pi_C, [\beta\gamma]_2) \cdot e([\beta\gamma]_1, \pi_B)$.
5. $e(\mathsf{PI}(x) + \pi_A, \pi_B) = e(\pi_C, g_2) \cdot e(\pi_H, [Z(\tau)\rho_A\rho_B]_2)$.

2 Attack Description

Note that the elements $\left\{\mathsf{pk}'_{A,i}\right\}_{i \in [0..n]}$ are not used at all by the verifier and honest prover, and thus could have been omitted from the key. We show that these elements allow a malicious prover to replace the public input arbitrarily when starting from a valid proof. Loosely speaking, we do this by adding a factor to π_A that "switches" the public input the proof is arguing about. The first verifier check – the "knowledge check" for π_A – should catch this, but the redundant elements allow the malicious prover to add the analogous factor to π'_A and pass the check. Details follow.

Suppose we are given a valid proof $\pi = (\pi_A, \pi_B, \pi_C, \pi'_A, \pi'_B, \pi'_C, \pi_H, \pi_K)$ for a public input $x = (x_1, \ldots, x_n) \in \mathbb{F}^n$. Choose any $x' = (x'_1, \ldots, x'_n) \in \mathbb{F}^n$.

Set

$$\eta_A := \pi_A + \sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}$$

$$\eta'_A := \pi'_A + \sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}$$

We claim that $\pi^* := (\eta_A, \pi_B, \pi_C, \eta'_A, \pi'_B, \pi'_C, \pi_K, \pi_H)$ is a valid proof for public input x'. The verifier checks with public input x' and proof π^* are

1. $e(\eta'_A, g_2) = e(\eta_A, [\alpha_A]_2)$.
2. $e(\pi'_B, g_2) = e([\alpha_B]_1, \pi_B)$.
3. $e(\pi'_C, g_2) = e(\pi_C, [\alpha_C]_2)$.
4. $e(\pi_K, [\gamma]_2) = e(\mathsf{PI}(x') + \eta_A + \pi_C, [\beta\gamma]_2) \cdot e([\beta\gamma]_1, \pi_B)$.
5. $e(\mathsf{PI}(x') + \eta_A, \pi_B) = e(\pi_C, g_2) \cdot e(\pi_H, [Z(\tau)\rho_A\rho_B]_2)$.

We show that the five equations all hold.

1. The check $e(\eta'_A, g_2) = e(\eta_A, [\alpha_A]_2)$; this is where the redundant elements crucially come into play.

   We have

   $$\eta'_A = \pi'_A + \sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}$$

   So

   $$e(\eta'_A, g_2) = e(\pi'_A, g_2) \cdot e\left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}, g_2\right)$$

   Since π is a valid proof, this is:

   $$= e(\pi_A, [\alpha_A]_2) \cdot e\left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}, g_2\right)$$

   Using $\mathsf{pk}'_{A,i} = \alpha_A \cdot \mathsf{pk}_{A,i}$ for every i,

   $$= e(\pi_A, [\alpha_A]_2) \cdot e\left(\alpha_A \cdot \left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}\right), g_2\right)$$

   Using bi-linearity of the pairing:

   $$= e(\pi_A, [\alpha_A]_2) \cdot e\left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}, [\alpha_A]_2\right)$$

   $$= e\left(\pi_A + \sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}, [\alpha_A]_2\right) = e(\eta_A, [\alpha_A]_2).$$

2. The second and third checks involve the unchanged $\pi_B, \pi'_B, \pi_C, \pi'_C$ and thus pass since π was valid.

3. The fourth and fifth equations are also identical in π and π^*. The only difference is that the latter replaces the term $\mathsf{PI}(x) + \pi_A$ with $\mathsf{PI}(x') + \eta_A$. And

$$\mathsf{PI}(x)+\pi_A = \mathsf{pk}_{A,0}+\sum_{i=1}^{n} x_i\mathsf{pk}_{A,i}+\pi_A = \mathsf{pk}_{A,0}+\sum_{i=1}^{n} x'_i\mathsf{pk}_{A,i}+\sum_{i=1}^{n}(x_i-x'_i)\mathsf{pk}_{A,i}+\pi_A = \mathsf{PI}(x')+\eta_A.$$

3 Security Proof in the Generic Group Model

Let us denote by BCTV' the scheme identical to the one in [BCTV14] described in Subsection 1.3, with two modifications:

- The elements $\left\{\mathsf{pk}_{A',i}\right\}_{i\in[0..n]}$ are excluded from the proving key.
- The scheme is only defined for QAPs where the polynomials $\{A_i\}$ satisfy
  1. The polynomials $\{A_i\}_{i\in[0..n]}$ are linearly independent (this condition already appears in [BCTV14] at [Par15]'s suggestion).
  2. $\mathrm{Span}_{\mathbb{F}}\left(\{A_i\}_{i\in[0..n]}\right) \cap \mathrm{Span}_{\mathbb{F}}\left(\{A_i\}_{i\in[n+1..m]}\right) = \{0\}$. [2]
- We also assume, mainly as a convenience, that $Z \in \{C_i\}_{i\in[n+1..m]}$. This is always the case if we want zero-knowledge, and the BCTV construction adds Z to $\{A_i\}, \{B_i\}, \{C_i\}$.

Remark 3.1. *We emphasize that the linear disjointness condition above is not just an artefact of the proof: The attack of the previous section can be carried out also without the redundant proving key elements, for public inputs x, x' such that $\mathsf{PI}(x) - \mathsf{PI}(x') \in \mathrm{Span}_{\mathbb{F}}\left(\{A_i\}_{i\in[n+1..m]}\right)$.*

We show BCTV' is knowledge sound in the generic group model. We will use the following simple linear algebra claim.

Claim 3.2. *Fix positive integers $n < m$. Let $v_0, \ldots, v_m$ be vectors in a vector space over $\mathbb{F}$ such that*

1. *$v_0, \ldots, v_n$ are linearly independent.*
2. *Defining $V = \mathrm{Span}_{\mathbb{F}}(v_0, \ldots, v_n)$ and $U = \mathrm{Span}_{\mathbb{F}}(v_{n+1}, \ldots, v_m)$, we have $V \cap U = \{0\}$.*

Suppose we are given $(x_0, \ldots, x_m), (a_0, \ldots, a_m) \in \mathbb{F}^m$ such that $\sum_{i=0}^m x_i \cdot v_i = \sum_{i=0}^m a_i \cdot v_i$. Then $(x_0, \ldots, x_n) = (a_0, \ldots, a_n)$.

Proof. Since $\sum_{i=0}^m x_i \cdot v_i = \sum_{i=0}^m a_i \cdot v_i$, we have

$$\sum_{i=0}^{n}(x_i - a_i)v_i = \sum_{i=n+1}^{m}(a_i - x_i)v_i$$

[2] In conversation with Alessandro Chiesa and Madars Virza, we learned that they were aware of the necessity of this condition, and it is satisfied by any QAP constructed in libsnark[lib]. It also already appears in [BGG17].

The zero-intersection condition implies $\sum_{i=0}^{n}(x_i - a_i)v_i = 0$ and the linear independence of $v_0, \ldots, v_n$ now implies $x_i - a_i = 0$ for $i \in [0..n]$.

□

We proceed to prove knowledge soundness of BCTV' in the generic group model. Following [Gro16] (see also Section 2 of [BG18] for formal details, and [BCI+12] for a more general treatment of this framework), it suffices to show knowledge soundness of BCTV' as a Non-Interactive Linear Proof (NILP). That is,

- The CRS elements are the field elements encoded in the SNARK CRS group elements, rather than the group elements.
- The malicious prover must output, given only the public input $x \in \mathbb{F}^n$, $\pi = (\pi_A, \pi_B, \pi_C, \pi'_A, \pi'_B, \pi'_C, \pi_H, \pi_K)$ as linear combinations of the CRS elements, such that the verifier equations hold as a polynomial identity in $\tau, \rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \gamma, \beta$ as formal variables.
- To prove knowledge soundness, it then suffices to efficiently derive from the coefficients of these linear combinations a QAP witness for x.

For BCTV', the NILP CRS is now the set of *field elements*

1. For $i \in [0..d]$, τ^i.
2. For $i \in [0..m]$
   (a) $\rho_A A_i(\tau)$
   (b) $\rho_B B_i(\tau)$
   (c) $\alpha_B \rho_B B_i(\tau)$
   (d) $\rho_A \rho_B C_i(\tau)$
   (e) $\alpha_C \rho_A \rho_B C_i(\tau)$
   (f) $\beta(\rho_A A_i(\tau) + \rho_B B_i(\tau) + \rho_A \rho_B C_i(\tau))$
3. For $i \in [n+1..m]$, $\alpha_A \rho_A A_i(\tau)$.
4. $\alpha_A, \alpha_B, \alpha_C, \gamma, \beta\gamma, \rho_a \rho_b Z(\tau)$.

and our verification equations are now:

1. $\pi'_A = \alpha_A \cdot \pi_A$.
2. $\pi'_B = \alpha_B \cdot \pi_B$.
3. $\pi'_C = \alpha_C \cdot \pi_C$.
4. $\gamma \cdot \pi_K = \beta\gamma \cdot (\mathsf{PI}(x) + \pi_A + \pi_B + \pi_C)$.
5. $(\mathsf{PI}(x) + \pi_A) \cdot \pi_B = \pi_C + \pi_H \cdot Z(\tau)\rho_A\rho_B$,

where

$$\mathsf{PI}(x) := \rho_A A_0(\tau) + \sum_{i=1}^{n} x_i \cdot \rho_A A_i(\tau).$$

Suppose now a prover has indeed output a valid proof $\pi = (\pi_A, \pi_B, \pi_C, \pi'_A, \pi'_B, \pi'_C, \pi_H, \pi_K)$ as linear combinations of the CRS elements, such that the verifier equations hold as polynomial identities. We show how to derive a QAP witness for x from the coefficients of these linear combinations. It will be convenient here to think of the proof elements as polynomials only in $\rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \beta, \gamma$ over the ring $\mathbb{F}[\tau]$; i.e. think of the coefficients of these polynomials (not to be confused with the coefficients of the linear combinations of CRS elements) as polynomials in τ.

The first equation implies that the linear combination for π'_A only has non-zero coefficients for elements with an α_A component, i.e.

$$\pi'_A = \sum_{i=n+1}^{m} a_i \cdot \alpha_A \rho_A A_i(\tau) + a^* \cdot \alpha_A$$

for some $a_{n+1}, \ldots, a_m, a^* \in \mathbb{F}$. Which implies

$$\pi_A = \sum_{i=n+1}^{m} a_i \cdot \rho_A A_i(\tau) + a^*$$

From the second and third checks, we can similarly conclude:

$$\pi_B = \sum_{i=0}^{m} b_i \cdot \rho_B B_i(\tau) + b^*$$

$$\pi_C = \sum_{i=0}^{m} c_i \cdot \rho_A \rho_B C_i(\tau) + c^*$$

Similarly, the fourth equation tells us that π_K must only use with non-zero coefficient elements with a β factor, i.e. the elements $\left\{\gamma\beta, \{\beta(\rho_A A_i(\tau) + \rho_B B_i(\tau) + \rho_A \rho_B C_i(\tau))\}_{i \in [0..m]}\right\}$. Hence we can write

$$\pi_K = \sum_{i=0}^{m} k_i \cdot \beta(\rho_A A_i(\tau) + \rho_B B_i(\tau) + \rho_A \rho_B C_i(\tau)) + k^* \cdot \gamma\beta$$

and therefore

$$\mathsf{PI}(x) + \pi_A + \pi_B + \pi_C = \sum_{i=0}^{m} k_i \cdot (\rho_A A_i(\tau) + \rho_B B_i(\tau) + \rho_A \rho_B C_i(\tau)) + k^* \cdot \gamma. \tag{1}$$

Looking at the ρ_A coefficient on both sides of (1), we have

$$\sum_{i=0}^{n} x_i \cdot \rho_A A_i(\tau) + \sum_{i=n+1}^{m} a_i \cdot \rho_A A_i(\tau) = \sum_{i=0}^{m} k_i \cdot \rho_A A_i(\tau),$$

and so

$$\sum_{i=0}^{n} x_i \cdot A_i(\tau) + \sum_{i=n+1}^{m} a_i \cdot A_i(\tau) = \sum_{i=0}^{m} k_i \cdot A_i(\tau).$$

Invoking Claim 3.2, this implies $k_i = x_i$ for $i \in [0..n]$. Since the coefficients of $\rho_B, \rho_A\rho_B$ must also match, we have

$$\sum_{i=0}^{m} b_i \cdot \rho_B B_i(\tau) = \sum_{i=0}^{m} k_i \cdot \rho_B B_i(\tau), \sum_{i=0}^{m} c_i \cdot \rho_A\rho_B C_i(\tau) = \sum_{i=0}^{m} k_i \cdot \rho_A\rho_B C_i(\tau)$$

So

$$\mathsf{PI}(x) + \pi_A = \sum_{i=0}^{m} k_i \cdot \rho_A A_i(\tau) + a^*, \pi_B = \sum_{i=0}^{m} k_i \cdot \rho_B B_i(\tau) + b^*, \pi_C = \sum_{i=0}^{m} k_i \cdot \rho_A\rho_B C_i(\tau) + c^*$$

Now invoking the fifth equation we have

$$(\mathsf{PI}(x) + \pi_A) \cdot \pi_B = \pi_C + \pi_H \cdot Z(\tau)\rho_A\rho_B. \tag{2}$$

Denote by $H(\tau)$ the degree at most d polynomial which is the part of π_H involving the terms $\{\tau^i\}_{i\in[0..d]}$ (inspection shows in fact $\pi_H = H(\tau)$, but this is not crucial for us). The $\rho_A\rho_B$ coefficient of each side in (2), giving

$$\left(\sum_{i=0}^{m} k_i \cdot A_i(\tau)\right)\left(\sum_{i=0}^{m} k_i \cdot B_i(\tau)\right) = \sum_{i=0}^{m} k_i \cdot C_i(\tau) + H(\tau)Z(\tau)$$

Since $k_i = x_i$ for $i \in [0..n]$, this means $k_{n+1}, \ldots, k_m$ is a QAP witness for public input x.

Acknowledgements

We thank Sean Bowe and Alessandro Chiesa for useful discussions and a review of this note.

References

[BBFR15] M. Backes, M. Barbosa, D. Fiore, and R. M. Reischuk. ADSNARK: nearly practical and privacy-preserving proofs on authenticated data. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 271–286, 2015.

[BCI+12] N. Bitansky, A. Chiesa, Y. Ishai, R. Ostrovsky, and O. Paneth. Succinct non-interactive arguments via linear interactive proofs. *IACR Cryptology ePrint Archive*, 2012:718, 2012.

[BCTV14] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014.*, pages 781–796, 2014.

[BG18] S. Bowe and A. Gabizon. Making Groth's zk-SNARK simulation extractable in the random oracle model. *IACR Cryptology ePrint Archive*, 2018:187, 2018.

[BGG17] S. Bowe, A. Gabizon, and M. D. Green. A multi-party protocol for constructing the public parameters of the pinocchio zk-snark. *IACR Cryptology ePrint Archive*, 2017:602, 2017.

[Fuc18] G. Fuchsbauer. Subversion-zero-knowledge snarks. In *Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part I*, pages 315–347, 2018.

[GGPR13] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct nizks without pcps. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 626–645, 2013.

[Gro16] J. Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 305–326, 2016.

[lib] https://github.com/scipr-lab/libsnark.

[Par15] B. Parno. A note on the unsoundness of vntinyram's SNARK. *IACR Cryptology ePrint Archive*, 2015:437, 2015.

[PHGR16] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: nearly practical verifiable computation. *Commun. ACM*, 59(2):103–112, 2016.

[sna] https://github.com/iden3/snarkjs.